cc-68-26(R)

OSCAR: A User's Manual With Examples (Revised)

by Jo Ann Baughman
Mary Berryman
and Joel Davis

November, 1968

COMPUTER CENTER
Oregon State University
Corvallis, Oregon 97331

CSCAR: A User's Manual with Examples (Revised)

November 1968

cc-68-26(R)

by: Jo Ann Baughman
Mary Berryman
and Joel Davis

Computer Center
Oregon State University
Corvallis, Oregon 97331

ACKNOWLEDGMENT

We wish to acknowledge the support of the National Science Foundation grant, GJ-51, NSF Aufenkamp 370 Regional Center, which aided the development of this manual.

TABLE OF CONTENTS

(PART TWO is a revision of "Several Illustrated Examples in OSCAR," by Lise Hedberg and Joel Davis.)

TABLE OF CONTENTS - Continued

TABLE OF CONTENTS - Continued

Page

PART THREE

SECTION A: (Continued)

TABLE OF CONTENTS - Continued

Page

PART THREE

[**NOTE**: See Reference 7, Appendix F, for a complete listing of & commands.]

OSCAR: A User's Manual With Examples

PART ONE

Introduction

OSCAR (Oregon State Conversational Aid to Research) is an arithmetic interpreter which provides the user with a "conversational" mathematical programming language. OSCAR is available as one option of the OS-3 operating system for use at remote teletypewriter consoles which are connected to the CDC 3300 by telephone cable. It provides the user with direct access to the CDC 3300 computer to solve mathematical problems in a "conversational" mode. That is, the user types a statement (or set of statements), signals the computer that he wants an answer, and the computer immediately types a response to the statement(s). These statements may be relatively simple mathematical expressions, or they may involve more complicated calculations involving several steps. It is also possible to group several statements together to be executed or solved as part of a larger, more complex problem. In either case, the user has access to what is, operationally, a very large high-speed calculator to perform mathematical or logical operations required to solve the particular user problem.

OSCAR performs in two modes: direct and indirect mode. In DIRECT MODE, OSCAR functions as a sophisticated desk calculator and evaluates mathematical expressions one at a time as the user enters them at the teletype. The user types

a statement and the computer then responds on the teletype. After the response has been typed out, the statement cannot be executed again unless it is retyped in its entirety. An example of a direct mode instruction would be SET X=6; Y=7; PRINT X*Y (ESC). This instruction is made up of several parts separated by semi-colons and when the escape key (ESC) is depressed the line is executed part by part. In order to execute the line again it must be retyped. In INDIRECT MODE, OSCAR allows sets of numbered statements to be stored so that it is possible to execute these sets more than once without retyping them. This alleviates the problem of having to retype a set of instructions when the same computations must be performed on several sets of data. An example of the indirect mode stored program is Example 3, Page 2.7.

To use OSCAR a job number and user number must first be obtained either from the Computer Center or from the instructor of the course involved. After the job number and user number have been obtained, the first thing that must be done to use OSCAR is to LOGON. In the following examples all information supplied by the OS-3 system will be underlined. To insure that the teletypewriter is, in fact, connected to the computer and that the OS-3 system is operating, the user should type the character "A" while holding the control shift key depressed (hereafter referred to as control shift A). The teletypewriter will respond with a pound sign (#) which indicates that the system is "on-line" and operating. If the pound sign was

typed on the teletypewriter, then the user may "logon" by typing his job number, a comma, his user identification, and a carriage return (CR). After the teletypewriter types another pound sign the user types the word OSCAR. The teletypewriter will then type OSCAR AT YOUR SERVICE and you are ready to begin. Diagramatically, this sequence appears as follows:

/CONTROL SHIFT A/

Job number, User IDENTIFICATION (CR)

OSCAR (CR)

OSCAR AT YOUR SERVICE V36 09/26/68 1502

Now the user is ready to enter instructions available under OSCAR which will perform the operations necessary to solve the particular user problem.

When the user is finished using OSCAR he may return to the OS-3 control mode by typing /CONTROL SHIFT A/. The user then has the option of utilizing some other feature of the OS-3 operating system, recalling OSCAR, or logging off. If the user chooses to log off he should type the word LOGOFF (or LOGOUT) and a carriage return (CR).

In PART TWO, the main part of each example is a sample printout from the teletypewriter. This sample is augmented by indicating where non-printing keys are used (see Special Keys and Symbols, Page 1.3). In some cases, the example is preceeded by an introduction. Every example is followed by comments and explanations. The comments are referenced

to specific portions of each example by a number and a letter which correspond to the number-letter combinations which appear in the right margin of each sample printout.

In PART THREE, there are two sections. Section A is a list of functions and operators which are presently available or which will soon be implemented. These functions and operators are listed in alphabetical order. They are then defined, with examples given of their usage, brief explanatory comments on usage procedures given if needed, and in some cases, further references are included. Section B is a list of COMMANDS which may be used under OSCAR. The commands are also listed in alphabetical order with definitions and examples of usage accompanying them.

Special Keys and Symbols

In OSCAR, the following symbols are used for the arithmetic operations: addition (+), subtraction (-), multiplication (*), division (/), powers (↑) and square root (SQRT). Certain functions are available, e.g. sine (SIN) and cosine (COS), see Part Three. It should be emphasized that OSCAR is still being developed and more functions will soon be available, e.g. tangent (TAN), arctangent (ARCTAN), numerator, denominator, etc. It is also possible for the user to define functions and subroutines; this is shown in Part Two, Example 4.

Some keys have special meaning in OSCAR. They will be referred to with the following symbols:

Control Shift A	Type the character "A" while the control shift is depressed. This is used to interrupt OSCAR if it is calculating. This causes the computer to type #. The pound sign is the symbol for the OS-3 control mode.

EXAMPLE:

Control Shift A

MI CR

[NOTE: The command MI, manual interrupt, will return the user to OSCAR.]

CR The carriage return key terminates a line, returns control to the computer and causes a line feed and carriage return.

EXAMPLE:

| Control Shift A |

Job Number, user identification CR

OSCAR

OSCAR at your service

[NOTE: The underlined characters are printed by the computer.]

ESC The escape key terminates a line. Some keyboards will not have this key, in which case the "ALT MODE" key or a | Control Shift W | will do the same thing. These keys will terminate the line and return control to the computer. (The result is similar to a CR, but does not include the carriage return and line feed.)

LF The line feed key.

| BREAK | The break key is used to interrupt OSCAR if it is typing.

EXAMPLE:

BREAK | BREAK RELEASE | Control Shift A

MI CR

* A line starting with * or ending with * is ignored.

\ The backslash is obtained by typing L while depressing shift. The backslash causes the space or character that precedes it to be ignored.

& The ampersand is used in combination with certain commands.

EXAMPLE:

&PRECISION = 10

[NOTE: This command would change the precision from the standard 6 significant digit to the specified 10.]

&PROGRAM, 6, .1

[NOTE: This command can be used when creating stored programs. See Part Two, page 2.7, Example 3.]

::= The emphatic assignment will clear variables before assigning values to them.

EXAMPLE:

X = 'Print Y'

X::= 24 ↑ 2 + 15.1 + 10.1I

[NOTE: See CLEAR, Part Three, page 3A.3.]

...	The ellipses allow the user to extend a line without returning control to the computer. On the teletypewriter unit, "end of line" is indicated by a bell or a red light. At this signal the user types three periods, OSCAR provides carriage return and line feed and the user can then type the continuation of the line. EXAMPLE: A = ARRAY ((1 2 3 4)(5 6 ... 7 8)(9 10 11))
%	The per cent symbol is used to change the precision. See PREC, Part Three, page 3A.23.
<	The symbol that is accepted for the relational operator "less than" (LSS may also be used. See Part Three, page 3A.15.)

$>$	The symbol that is accepted for the relational operator "greater than." (GTR may also be used. See page 3A.11, GTR.)
= or ←	Either of these symbols can be used for assignment.

[NOTE: For a more complete list of symbols, See Appendix B, <u>Special Symbols in OSCAR</u> in "A Brief Description of OSCAR (Revised)," by Joel Davis, cc-68-24.]

References

This report is intended as an introduction to OSCAR, only. It does not contain a complete description of OSCAR, nor does it describe the time-sharing system, OS-3, under which OSCAR is operating. Users who want more information about OSCAR and the OS-3 system should refer to the following manuals and reports:

1. OS-3 User's Manual (Revised), by Gil Bachelor, cc-68-3.

2. Teletype Operation, Department of Civil Engineering, cc-68-6.

3. Computer Center User's Manual, by Ron Davis and Kay Porter, cc-68-14.

4. OS-3 Teletype Editor Manual (Revised), by Fred Dayton and Walter Massie, cc-68-17.

5. A Control Mode Manual for OS-3, Version 2.0, by Walter Massie, cc-68-21.

6. Fortran Manual for OS-3, Version 2.0, by Walter Massie, cc-68-22.

7. A Brief Description of OSCAR, Revised, by Joel Davis, cc-68-24.

8. Using the Plotter: Documentation and Examples, by Jo Ann Baughman and Dean Pielstick, cc-68-25

9. Free Form Input for OS-3 Fortran, by Walter Massie, cc-68-27.

10. OSU Computer Center Program Library Catalog, cc-68-28.

11. User's Manual for OS-3, by Walter Massie and G.A. Parks of the Civil Engineering Department, 1967.

12. SCANIN, A Free Format BCD to Floating Point Converter, by Walt Pawley, cc-68-10.

PART ONE

Introduction

Special Keys and Symbols

References

Example 1. Demonstration of simple operations.

```
Control Shift A
#job number,user identification  (CR)
#OSCAR  (CR)                                                         1a
OSCAR AT YOUR SERVICE V18 03/11/68   1650
*EXAMPLE 1  (CR)                                                     1b
X=6;Y=7;X*Z\Y IS       (ESC)    42
                                                                     1c
X←6;Y←7;PRINT X*Y      (ESC)    42
6/7 IS    (ESC)    6/7
5/10 IS   (ESC)    1/2                                               1d
6./7. IS    (ESC)        0.857143
6P10/7P10 IS  (ESC)      0.8571428571
LET F(X,Y)=X*Y-5*X+3; F(4,7) IS      (ESC)     11                    1e
C::='A↑2+B↑2'   (CR)     [NOTE:  The double colon, which would
                                 clear any previous value from C,
A=1;B=2;PRINT C  (ESC)    5      is not necessary in this case.      1f
                                 See CLEAR, page 3A.3.]
B=3;PRINT C  (ESC)       10
FOR X=2 BY 2 TO 10 PRINT X,X↑2,X↑3   (ESC)     2        4...
       8        4       16       64      6     36      216...
       8       64      512       10    100   1000

FOR X=2 BY 2 TO 10 PRINT X,X↑2,X↑3,CR   (CR)
       2        4        8
       4       16       64
       6       36      216
       8       64      512                                           1g
      10      100     1000

FOR X=0.2 BY 0.2 TO 1.0 PRINT X,X↑2,SQRT(X),CR   (CR)
       0.2              0.04             0.4472135
       0.4              0.16             0.6324555
       0.6              0.36             0.7745965
       0.8              0.64             0.894427
       1.000000         1.000000         1.000000
LOGOFF  (CR)
Control Shift A
#LOGOFF   (CR)                                                       1h
TIME     2.837 SECONDS.   MFBLKS 0000   COST $
#
```

Comments to Example 1

1a OSCAR is available when the CDC 3300 computer is running under the OS-3 operating system. To LOGON the user needs a job number and user identification registered with the OS-3 system. Following the proper LOGON, the computer types #, the user types OSCAR and depresses the carriage return key. The computer types OSCAR AT YOUR SERVICE, etc., and is ready to receive instructions.

1b A line starting with an asterisk (*) is ignored by the computer and may be used as a comment. The same is true of a line ending with an asterisk. A single character is cancelled by a backslash (\), see 1c. Two of these will cancel the last two characters typed.

1c The two statements X=6;Y=7;X*Y IS and X←6;Y←7;PRINT X*Y are equivalent. In each case, 6 replaces X. 7 replaces Y and X*Y is calculated and printed. The backslash cancels the Z typed by mistake. Several statements on a line are separated by semicolons.

1d A simple fraction is left unchanged or reduced when possible. Division of two decimal numbers will give the quotient with six figures unless a different precision is indicated. With precision 10 (6P10/7P10) the quotient is given with ten figures.

1e The word LET enables the user to define functions.

1f C is defined as the literal expression A^2+B^2, which must be enclosed in apostrophes. The two colons following C clear the left-hand side. When both A and B are defined, C may be calculated and printed. When only B is redefined, A retains the original value.

1g The FOR statement is used to define a series of operations. Here X^2 and X^3 are calculated for X=2,4,6,8,10. If no carriage return is indicated, several numbers are printed on a line. The ellipses indicate a continuation of the line. Ellipses may also be used in input to extend a line (see comment 6b, page 2.26.) The letters CR included in the print statement will cause a carriage return at that point. The numbers and the intervals in the FOR statement may be decimal numbers or complex numbers as well as whole numbers.

1h The first LOGOFF had no effect. It is necessary to use $\boxed{\text{Control Shift A}}$ and wait for the computer to print # before the LOGOFF command can be used. MFBLKS stands for memory file blocks. Only when the user is working with files, as in Example 5, will the number be different from zero. TIME indicates to the nearest millisecond the computer time used in performing the calculations.

Example 2. Functions and constants.

```
Control Shift A

#job number,user identification   CR

#OSCAR   CR

OSCAR AT YOUR SERVICE V18 03/11/68  1729

*EXAMPLE 2   CR

SQRT(2) IS   ESC ...
       1.4142135623730950488016887242096980785696718753770

SQRT(2.) IS     ESC      1.414215                                2a

SQRT(2P14) IS   ESC      1.4142135623731

PRINT PI        ESC      3.1415926535898

PRINT E         ESC      2.7182818284590                         2b

SIN(2) IS       ESC      0.90929742682568

COS(2) IS       ESC     -0.41614683654714З

SIN(2.) IS      ESC      0.909296                                2c

SIN(PI/6) IS    ESC      0.50000000000000

SIN(PI/12) IS   ESC      0.25881904510252

CONST=PI/180;PRINT CONST      ESC      0.017453292519943

SIN(15*CONST) IS   ESC   0.25881904510252                        2d

COS(15*CONST) IS   ESC   0.96592582628906

TAN(15*CONST) IS   ESC   0.26794919243112

LOG(4) IS          ESC   1.3862943611240                         2e

EXP(2) IS          ESC   7.3890560989341

Control Shift A

#LOGOFF   CR
TIME    2.830 SECONDS.  MFBLKS 0000
#
```

Comments to Example 2

2a An integer with no decimal point is considered an exact number. The square root of an exact number is given with 50 digits; this is maximum precision. A number with a decimal point and no precision indicated is assumed to have precision 6. The square root of such a number will have approximately the same precision — 6 or 7 digits will be printed. When precision 14 is indicated for the argument, 14 or 15 digits will be printed.

2b The constants π and e are preset with 14 digits.

2c The argument of the sine or cosine function is an angle given in radians. The sine and cosine functions are accurate to about precision 14 or 15. Sine and cosine of exact numbers will be given with 14 or 15 digits. Sine or cosine of a decimal number will have about the same precision as the argument — up to 14 or 15 digits.

2d To calculate sine or cosine of angles given in degrees it is necessary to multiply the number of degrees with a constant, π/180. Here, this constant is calculated and given the name CONST, for convenience sake only. OSCAR recognizes variable names up to four characters. When more characters are used, they are ignored.

2e The function TANGENT has not yet been implemented. A call for such a function will cause an error statement as seen here. Functions not available may be defined by the user on an individual basis, as is shown in Example 5.

Example 3. Stored Program.

```
#job number,user identification   CR

#OSCAR   CR

OSCAR AT YOUR SERVICE V35 07/25/68  0919

&PROGRAM,1,.01   CR        [NOTE: A stored program can also be        3a
                                  entered by another method
1.01:READ X   CR                  whereby the user types the
                                  sequence numbers (1.01: in
1.02:Y=X   CR                     this example).]

1.03:Z::='Y↑2+16*Y+4.5'   CR

1.04:PRINT "Z",Z,CR   CR

1.05:PRINT "SQRT(Z)",SQRT Z,CR   CR                                    3b

1.06:PRINT "END OF PART ONE   CR

1.07:   ESC

DO PART 1   CR
3   CR                                                                 3c
Z                  61.50000
SQRT(Z)             7.842194
END OF PART ONE

DO PART 1   CR
-89.76   CR                                                            3d
Z                6625.2
SQRT(Z)            81.39535
END OF PART ONE

2.2:S=Z   CR

2.3:LET D=S-2I+89.5*PI   CR
                                                                       3e
2.5:PRINT "D IS",D,CR   CR

2.6:PRINT "END OF PART TWO   CR

DO PART 1   CR
4   CR
Z                  84.50000
SQRT(Z)             9.1923882
END OF PART ONE
                                                                       3f
DO PART 2   CR
D IS              365.673-2I
END OF PART TWO
```

Example 3: Continued

```
2.7:2.9:PRINT "2.9"   CR                    } 3g
2.8:PRINT "2.8"   CR

DO PART 1   CR
3   CR
Z                    61.50000
SQRT(Z)               7.842194
END OF PART ONE

DO PART 2   CR                              } 3h
D IS                342.673-2I
END OF PART TWO

2.8

2.9
```

Control Shift A

```
#TIME   CR
TIME 7.611 SECONDS.   MFBLKS 0   CFBLKS 0    } 3i
#TRAFFIC   CR
TRAFFIC IS   12
#LOGOFF   CR
TIME 7.700 SECONDS.   MFBLKS 0
```

Comments to Example 3

3a The command &PROGRAM, <Part Number>.<Step Number> (e.g. &PROGRAM, 1.01) will cause the sequence number <Part Number>.<Step Number> (1.01 in the example) to be printed by the computer. Each number cannot be larger than 999999. The user can then type the contents of that line and terminate the line with a CR . The contents of the line 1.01 will be placed in storage identified by the number. The computer will then type the next sequence number. This will continue until ESC is pushed.

3b The stored program can also be set up with the user's typing the sequence numbers. The contents of line 1.01 will be put in storage identified by that number.

3c The command DO PART 1 will cause all of the text found in numbers starting with one (1) to be executed. The command READ X will cause OSCAR to expect a value for X to be printed by the user. The value in this case is three (3). It will be put in a storage location identified by X. See READ, page 3A.25.

3d PART 1 is executed again, this time with the value -89.76 being read in.

3e Another stored program has been typed in and the text will be in storage identified by the sequence numbers.

3f PART 1 is executed, this time with the value 4 for X. PART 2 depends on the value of Z found in PART 1. PART 2 is executed.

3g Two more statements are added to PART 2. In this case, statement 2.7 contains 2.9. When 2.7 is executed, the contents (2.9:PRINT 2.9) will be stored under 2.9. The statements will be executed in the order of their sequence numbers. See part 3h.

3h PART 1 is executed with the value 3 being read in for X. PART 2 is executed using the value of Z calculated in PART 1.

3i The command TIME (used in control mode #) will cause the number of seconds used since the user logged on to be printed, along with the number of file blocks (MFBLKS) used. The command TRAFFIC (used in control mode #) will cause the number of users to be printed. (See Reference 5, page 39.01.)

Example 4. Vectors and matrices.

```
#job number,user identification   CR

#OSCAR   CR

OSCAR AT YOUR SERVICE V18 04/03/68  0853

A←ARRAY(1 2 3);B←ARRAY(7 6 5);PRINT A+B,A-B,A*B   ESC      4a
 (8  8   8  )
 (-6  -4  -2  )                 34

A←ARRAY(1 2 3 4 5 6 7 8 9);PI\RINT A   ESC
 (1 2 3 4 5 6 7 8 9 )

A(3)←10;PRINT A   ESC
 (1 2 3 10 4 5 6 7 8 9 )

A(7:9)←0;PRINT A   ESC                                     4b
 (1 2 10 4 5 6 0 0 0  )

CLEAR A(7:9);PRINT A   ESC
 (1 2 10 4 5 6   )

FOR I=4 TO 6 CLEAR A(I); PRINT A   ESC
 (1 2 10 )

A←ARRAY((1 2 3)(4 5 6)(7 8 9))   ESC

PRINT A(2)   ESC                                           4c
 (4 5 6 )

A(2,3)←0;PRINT A   ESC
 ( (1  2  3  )
   (4  5  0  )
   (7  8  9  ) )

PRINT A*A   ESC                                            4d
 ( (30  36  30 )
   (24  33  12 )
   (102 126    102   )    )

A(2)←ARRAY(10 11 12);PRINT A   ESC
 ( (1   2   3  )
   (10  11  12 )
   (7   8   9  )   )                                       4e

CLEAR A(2);PRINT A   ESC
 (   (1  2  3  )  []
     (7  8  9  )   )

CLEAR A(3);PRINT A   ESC
 (   (1  2  3  )  )
```

Example 4. Continued

Control Shift A

```
#LOGOFF   CR
TIME  2.561 SECONDS.  MFBLKS 0000   COST $
#
```

Comments to Example 4

4a A vector is defined using the word ARRAY. The vector components are enclosed in parentheses and separated by spaces. Commas may also be used to separate elements of arrays. Here A and B are defined as two vectors and their sum, difference, and product are calculated.

4b A is defined as a vector with nine components. Component number N is referred to as A(N). The statement A(3)←10 changes A(3) to 10 leaving the other components unchanged. When several successive components are changed to the same value the colon may be used as in A(7:9)←0. The statement CLEAR A(7:9) clears the last three components. The FOR statement may also be used to clear successive elements.

4c A matrix is also defined using the word ARRAY. The matrix is defined row by row, the elements of each row are enclosed in parentheses and outer parentheses

enclose all rows. A is here defined as a 3 by 3 matrix. A(2) refers to the second row, A(2,3) to the third element in the second row, etc.

4d The operations addition, subtraction and multiplication of matrices are indicated in the same way as for vectors (see 4a) only multiplication is performed here.

4e Any of the rows may be redefined leaving the other rows unchanged. The statement CLEAR A(2) leaves row two undefined; it does not affect row three. When row three is cleared also, only row one is printed. The print statement will include all rows up to the last one which contains actual values.

[NOTE: The elements of vectors and matrices may be complex as well as real numbers. In general, any type of number legal in OSCAR is legal as an array element. The maximum size of a one-dimensional array is 255. A two-dimensional array may be 255x255, a three-dimensional one 255x255x255, etc., limited only by the size of the computer memory. Matrices may have any number of dimensions.]

EXAMPLE 5. Subroutine Files: Subroutines to calculate the transpose and the inverse of a matrix.

Certain subroutines, e.g. the sine and cosine subroutines, are already part of the OSCAR subroutine library and are available to all users. In addition, it may be convenient for a user to establish subroutine files of his own related to the type of calculations that are of special interest to him. It is not possible to establish subroutine files directly from OSCAR. The user may, however, take advantage of the fact that OSCAR is part of the OS-3 operating system. Files established under the OS-3 system may be called from OSCAR and used as subroutines. In this example two subroutines are established. The first, called TRANS(M,N,AA), generates the transpose of a matrix AA, with M rows and N columns. The second, called INV(N,AA), calculates the inverse of a square matrix AA, of order N, by Jordan's method.

The example has been divided into three parts:

1. Entering the subroutines on files,
2. Using the subroutines in calculations in OSCAR, and
3. Making a paper tape copy of one of the subroutines.

The subroutines are entered by the aid of the OS-3 library routine EDIT. The first subroutine is typed in on the keyboard of the teletypewriter; the second is read from paper tape. The paper tape had been punched as a copy of another user's file, as shown in Part III (of this example). This provides a means of transferring information between files of different users.

Example 5. Part I

```
#job number,user identification  (CR)

#EDIT   (CR)                                                    5a
]INPUT   (CR)
00001:LET TRANS(M,N,AA)=[CLEAR BB;FOR I=1 TO M DO TR;BB]   (CR)
00002:TR::='FOR J=1 TO N DO BB(J,I)←AA(I,J)'   (CR)
00003:   (ESC)
]LIST   (CR)                                                    5b
00001:LET TRANS(M,N,AA)=[CLEAR BB;FOR I=1 TO M DO TR;BB]
00002:TR::='FOR J=1 TO N DO BB(J,I)←AA(I,J)'
]OUT,TRANS   (CR)                                               5c

]  [Control Shift A]   (CR)

#FP,TRANS   (CR)

#EDIT   (CR)
]TAPE
                                                                5d
      [Insert tape in tape reader]  (CR)  [Start tape]
LET INV(N,AA)=[FOR I=1 TO N DO[BB=AA(I);AA(I,1:N)←0;AA(I,I)
=1;SS;AA]]
SS←'FOR J=1 TO N DO[TT←AA(J,I)/BB(I);AA(J)=AA(J)-TT*BB;AA
(J,I)←TT]'
]OUT,INV   (CR)

]RESEQ   (CR)                                                   5e

]LIST   (CR)
00001:
00002:
00003:LET INV(N,AA)=[FOR I=1 TO N DO[BB=AA(I);AA(I,1:N)←0;AA
(I,I)=1;SS;AA]]
00004:SS←'FOR J=1 TO N DO[TT←AA(J,I)/BB(I);AA(J)=AA(J)-TT*BB;
AA(J,I)←TT]'

]ERASE,1,2   (CR)

]RESEQ   (CR)
]LIST   (CR)
00001:LET INV(N,AA)=[FOR I=1 TO N DO[BB=AA(I);AA(I,1:N)←0;AA
(I,I)=1;SS;AA]]
00002:SS←'FOR J=1 TO N DO[TT←AA(J,I)/BB(I);AA(J)=AA(J)-TT*BB
;AA(J,I)←TT]'

]OUT,INV   (CR)

]  [Control Shift A]   (CR)                                     5f

#FP,INV   (CR)
#LOGOFF   (CR)
TIME    1.896 SECONDS.   MFBLKS    0001                          5g
#
```

Example 5. Part II

```
#job number,user identification   (CR)

#OSCAR   (CR)

OSCAR AT YOUR SERVICE V18 04/03/68  0911                              }
                                                                      }
&CONTROL,TRANS,TTY   (CR)                                             }5h
LET TRANS(M,N,AA)=[CLEAR BB;FOR I=1 TO M DO TR;BB]                    }
TR::='FOR J=1 TO N DO BB(J,I)←A(I,J)'                                 }

A←ARRAY((1 2 3)(4 5 6)); PRINT A   (ESC)
 (   (1   2   3   )
     (4   5   6   )   )

B=TRANS(2,3,A); PRINT B   (ESC)
 (   (1   4   )
     (2   5   )
     (3   6   )   )                                                    5i

PRINT A*B,CR,B*A   (ESC)
 (   (14   32   )
     (32   77   )   )

 (   (17   22   27   )
     (22   29   36   )
     (27   36   45   )   )

&CONTROL,INV,TTY   (CR)

LET INV(N,AA)=[FOR I=1 TO N DO[BB=AA(I);AA(I,1:N)←0;AA(I,I)=...       5j
1;SS;AA]]
SS←'FOR J=1 TO N DO[TT←AA(J,I)/BB(I);AA(J)=AA(J)-TT*BB;AA...
(J,I)←TT]'

A←ARRAY((50  107  36)(25  54  20)(31  66  21))   (CR)

PRINT A,CR,INV(3,A),CR,A*INV(3,A)   (CR)
 (   (50   107      36 )
     (25   54   20   )
     (31   66   21   )   )

 (   (-186   129   196   )                                             5k
     (95   -66   -100   )
     (-24   17   25   )   )

 (   (1   0   0   )
     (0   1   0   )
     (0   0   1   )   )

 Control Shift A
                                                                       5l
#LOGOFF   (CR)
TIME    3.531 SECONDS.   MFBLKS    0000
#
```

Example 5. Part III

```
#job number,user identification   CR

#COPY,IN=INV  Turn on tape punch    CR

   LET INV(N,AA)=[FOR I=1 TO N DO[BB=AA(I);AA(I,1:N)←
   0;AA(I,I)=1;SS;AA]]
   SS←'FOR J=1 TO N DO[TT←AA(J,I)/BB(I);AA(J)=AA(J)-TT*BB;
   AA(J,I)←TT]'

#LOGOFF   CR
TIME   0.099 SECONDS.  MFBLKS  0000
#
```

5p

Comments to Example 5

In all the previous examples, the only part of the OS-3 system used was the library routine OSCAR. In this example other features of the OS-3 system are used as well; in particular, the library routine EDIT. The OS-3 system and the library routine EDIT are described in the manuals listed on page 6. Brief descriptions of the statements used in this example are included in the comments.

The sign # is printed by the computer. It indicates that the computer is ready to accept OS-3 control statements, such as: LOGON, LOGOFF, EQUIP, FP, SAVE, or a call for an OS-3 system routine, such as: EDIT, OSCAR, COPY.

The sign] is also printed by the computer. It indicates that the computer is operating under EDIT in the command mode and is ready to accept EDIT commands, such as: INPUT, OUTPUT, LIST, and TAPE.

Control Shift A	is used to get back to the general OS-3 system from either EDIT or OSCAR.

5a A call for EDIT followed by the command INPUT prepares a work area in the computer memory. The computer provides a sequence number, the user types a line terminated by pressing the carriage return key, and the computer continues to supply sequence numbers until the escape key is pressed. As in OSCAR, backslash (\) is used to cancel single characters; however, (@) is used to cancel whole lines.

5b The command LIST will cause the computer to print the contents of the work area, and may be used to check the input.

5c The command OUT,TRANS equips a file, rewinds the file, and places the text under the name specified in the file directory under the name TRANS. File names may have up to 8 characters and need not necessarily be the name of the subroutine as is the case here. FP, meaning file protect, protects the file so that it will be available for use later. A file which is not file protected might be accidentally destroyed.

5d The EDIT routine is used for input of the second subroutine also. The command TAPE prepares a work area for input and reads paper tape. At the end of the tape input the escape key is pressed. At this point it is a good idea to put the text in the file directory with the command OUT,INV, (See 5c).

5e The statements in this section are included for checking purposes only. No sequence numbers are provided by the TAPE command. The command RESEQ (resequence) assigns a sequence number to each line in the work area. Without RESEQ, LIST would print the contents of the work area without sequence numbers. The two blank lines are

caused by two carriage returns punched at the beginning of the paper tape. They may be left in or erased as is done here.

5f The corrections must also be placed out in the file directory with the command OUT,INV. The second subroutine is then file protected.

5g MFBLKS stands for memory file blocks. The number printed is the maximum number of files in memory at any time. Two files were used in this example. When the first file was saved and file protected, it was taken out of memory and, as a result, only one file was in memory at a time and the number printed is 1. (See also 5l)

5h &CONTROL,TRANS,TTY is an OSCAR command which transfers the statements from the file specified to the OSCAR scratch area and executes them one at a time. (The ampersand is typed by the user.) TTY causes the statements to be printed on the teletypewriter. TTY may be omitted, in which case the statements are not printed, but are still executed. See page 3B.0 for &.

5i A is defined as a matrix with two rows and three columns. The name of the subroutine is TRANS(M,N,AA), where M is the number of rows, N the number of columns, and AA

the name of the matrix. B is defined as the transpose of A by the statement: B=TRANS(2,3,A). The multiplications A*B and B*A are equivalent to A*A' and A'*A.

5j &CONTROL,INV,TTY transfers and prints the statements on the file specified, in this case the subroutine INV(N,AA), which calculates the inverse of a square matrix AA of order N. A↑(-1) is also A^{-1}.

5k A is defined as a 3 by a 3 matrix. A^{-1} may be calculated as INV(3,A). The print statement includes A,A^{-1} and A*A^{-1}. The matrix elements are not restricted to whole numbers. (See NOTE on page 2.13.)

5l Although two files were used in this example, no memory blocks were used. The files are not copied into memory as such. The information is transferred to the OSCAR scratch area as instructions.

5m COPY is an OS-3 utility routine which may be used to copy information from a file. COPY,IN=INV causes the content of the file specified to be typed on the teletypewriter; by turning on the paper tape punch, the user gets a paper tape copy at the same time. A paper tape can also be generated by using the EDIT command TTP. (See Reference 4.)

EXAMPLE 6: Data Files

Data files may be written and read from OSCAR using the OSCAR commands, &OUTPUT,LUN and &INPUT,LUN, where LUN stands for logical unit number of an already equipped file. Following &OUTPUT,LUN, each PRINT or EPRINT statement will cause information to be written on the file specified by LUN. EPRINT (Emphatic PRINT) is a special print command which prints structures like arrays and alphabetical strings with the necessary punctuation to be read in again by OSCAR. Following &INPUT,LUN, each READ statement will read information from the file specified by LUN.

See Section B of Part Three of this manual for definitions and examples of & commands beginning on page 3B.1.

Example 6. Data Files.

```
#job number,user identification   CR                                6a
#OSCAR   CR

OSCAR AT YOUR SERVICE V18 04/19/68  1620
&EQUIP,1=FILE

A=101;B=202;C=303   CR                                              6b
D=ARRAY((1.11111 , 1.22222 , 1.33333 , 1.44444)(2.11111 , 2...
.22222 , 2.33333 , 2.44444))   CR
&OUTPUT,1,TTY   CR                                                  6c
PRINT "*RESULTS FROM TEST 1   ESC   *RESULTS FROM TEST 1            6d
PRINT A,B,C,CR   ESC   101   202   303                              6e
PRINT "*RESULTS FROM TEST 2   ESC   *RESULTS FROM TEST 2            6e
EPRINT D,CR   CR                                                    6d
 ARRAY...
 (  (1.11111 , 1.22222 , 1.33333 , 1.44444  )  ,...                 6e
    (2.11111 , 2.22222 , 2.33333 , 2.44444  )  )
&REWIND 1   CR
 Control Shift A                                                    6f
#COPY,IN=1   CR                                                     6g
  *RESULTS FROM TEST 1
       101      202      303
  *RESULTS FROM TEST 2
   ARRAY
   ( (1.11111 , 1.22222 , 1.33333 , 1.44444 )  ,           ...
     (2.11111 , 2.22222 , 2.33333 , 2.44444 )  )           ...

#OSCAR   CR

OSCAR AT YOUR SERVICE V18 04/19/68  1624                            6h
&REWIND 1   CR
&INPUT,1,TTY   CR

READ X   CR
*RESULTS FROM TEST 1
     101      202      303

X IS   ESC   101                                                    6i
&REWIND 1   CR

READ X,Y,Z   CR
*RESULTS FROM TEST 1
     101      202      303

X IS   ESC   101
Y IS   ESC   202
Z IS   ESC   303
```

Example 6. Continued

```
READ W   (CR)                                                       } 6j
*RESULTS FROM TEST 2
 ARRAY                                                  ...
 (  (1.11111 , 1.22222 , 1.33333 , 1.44444  )    ,     ...
    (2.11111 , 2.22222 , 2.33333 , 2.44444  )    )

W IS   (CR)
 (  (1.11111   1.22222   1.33333   1.44444  )
    (2.11111   2.22222   2.33333   2.44444  )    )
&OUTPUT,1   (CR)                                                    } 6k

EPRINT "RESULTS FROM TEST 3",CR   (CR)                              } 6l

PRINT X+Y,X+Z,Y+Z,CR   (CR)                                         } 6m

EPRINT "RESULTS FROM TEST 4",CR   (CR)                              } 6l

EPRINT 2*W,C   (CR)                                                 } 6m

&DATE   (CR)                                                        } 6n
```

Control Shift A

```
#SAVE,1=TEST   (CR)                                                 } 6o
#FP,1   (CR)
#LOGOFF   (CR)
TIME   3.478 SECONDS.   MFBLKS   0001
#

#job number,user identification   (CR)                             } 6p
#EQUIP,10=TEST   (CR)
#COPY,IN=10   (CR)
  *RESULTS FROM TEST 1
        101      202      303
  *RESULTS FROM TEST 2
   ARRAY                                                           ...
   ( (1.11111 , 1.22222 , 1.33333 , 1.44444 )    ,     ...         ...
     (2.11111 , 2.22222 , 2.33333 , 2.44444 )    )     ...
  "RESULTS FROM TEST 3"
        303      404      505
  "RESULTS FROM TEST 4"
   ARRAY                                               ...
   ( (2.22222 , 2.44444 , 2.66666 , 2.88888 )    ,     ...
     (4.22222 , 4.44444 , 4.66666 , 4.88888 )    )
  04/19/68  1628
```

Example 6. Continued

```
#REWIND 10    CR

#OSCAR   CR

OSCAR AT YOUR SERVICE V18 04/19/68  1631

&INPUT,10   CR

READ A,B,C,D,E,F,G,H,K,L,M    CR

READ ( M  )
   INPUT DATA ERROR.

A IS  ESC      101
B IS  ESC      202
C IS  ESC      303
D IS  ESC
 ( (1.11111  1.22222  1.33333  1.44444  )
   (2.11111  2.22222  2.33333  2.44444  )  )
E IS  ESC  RESULTS FROM TEST 3
F IS  ESC      303
G IS  ESC      404
H IS  ESC      505
K IS  ESC  RESULTS FROM TEST 4
L IS  ESC
 ( (2.22222  2.44444  2.66666  2.88888  )
   (4.22222  4.44444  4.66666  4.88888  )  )

M IS  ESC    []
```

$\boxed{\text{Control Shift A}}$

```
#LOGOFF   CR
TIME  1.783 SECONDS.  MFBLKS 0000
#
```

6q

6r

Comments to Example 6

In this example the OSCAR commands: &OUTPUT, &INPUT, &REWIND, and &DATE are used. In all these commands the ampersand can be typed by the user. This is not the case with the sign #. This sign is typed by the computer indicating that the computer is operating under the OS-3 system control mode. See page 3B.0 for &.

6a Files can be equipped directly from OSCAR. They also can be equipped outside OSCAR by the OS-3 control statement: EQUIP,LUN=FILE. (See Reference 5, page 16.01.)

6b A, B, and C are defined as three variables. D is defined as an array. The input of D extends over more than one line. On the teletypewriter unit, "end of line" is indicated by a bell or a red light. At this signal, the user types three periods, OSCAR provides carriage return and line feed, and the user types the continuation of the line. The line is terminated when the user presses the carriage return or the escape key.

6c &OUTPUT,1 is an OSCAR command. Following this command all outputs are written on file 1 until another &OUTPUT command is given. This may have the form: &OUTPUT,LUN, where LUN is the logical unit number of a different file, or &OUTPUT,TTY, where TTY refers to the teletypewriter. &OUTPUT,LUN,TTY directs the output both to a file and

to the teletypewriter. An error statement after &OUTPUT,LUN will cause the output to revert to the teletypewriter. See Section B of Part Three, &OUTPUT, page 3B.7.

6d Text as well as numbers may be written on the file. Two different types of PRINT statements may be used in connection with text. The one used here has the form: PRINT"*TEXT ESC ; in this case, the text is written on the file in the form of a comment which will be ignored by the READ statement. See also 61.

6e To write the values of the three variables A, B, and C on the file, either of the statements PRINT or EPRINT may be used. To write the array D on the file, it is necessary to use the statement EPRINT (Emphatic Print). (PRINT would cause the numbers to be written on the file in a form which could not be read in again.) The letters CR mark the end of line on the file. It is essential that the last PRINT statement end with the letters CR or part of the information will be lost. However, if control is switched back to the teletypewriter by an &OUTPUT,TTY command, a CR is supplied automatically.

6f &REWIND 1 is an OSCAR command. It is similar to the OS-3 control statement REWIND 1, which may be used after the sign #.

6g The OS-3 routine COPY,IN=1 is used to print all information written on file 1. This is not an essential part of the procedure. It is done only to check the contents of the file.

6h The information on file 1 may be read from OSCAR following the command &INPUT,1; TTY causes the information to be typed on the teletypewriter. Normally, it is omitted. The statement READ X will read all information from the first line on the file. This includes the comment: *RESULTS FROM TEST 1, as well as the numbers: 101 202 303. The value 101 is assigned to X. The numbers 202 and 303 are not assigned to any variables.

6i After the command &REWIND 1 the same line is read again; this time by the statement READ X,Y,Z. The three numbers are assigned to the three variables. The line was written on the file by the statement PRINT A,B,C,CR. Any three variable names may be used in the read statement. It would have been possible to read the information by three separate statements: READ X; READ Y; READ Z.

6j The statement READ W reads the comment as well as the array. The printout following the statement W IS shows that the components of the array have been properly assigned to W.

6k After the command OUTPUT,1 the user may continue to write information on file 1. Had this command been preceded or followed by the command REWIND 1, the new information would have been written from the beginning of the file and replaced the old information. As it is done here, the new information is added to what is already on the file.

6l Under 6d, the text was written on the file in the form of a comment. Here it is written in the form of an alphanumeric string enclosed in quotation marks. The EPRINT statement must be used. The proper form is: EPRINT"TEXT",CR. The text will be read into a separate variable. See 6r.

6m The numbers written on the file are combinations of the numbers used previously. The components of the array have twice the value of the components in the previous array.

6n The command DATE is used here to write the current date and time on the file.

6o The file is saved under the name TEST, and file protected.

Comments to Example 9

9a The EQUIP statement assigns logical unit number 1 to the file named RK. The OSCAR control statement &CONTROL,1,TTY transfers the statements written on file RK to the OSCAR scratch area as instructions to be executed. TTY, which causes the statements to be printed on the teletypewriter, may be omitted. The instructions are written as four abbreviation statements, where the words RKSTEP, RK2, RKPROGRAM, and STEPANDPRINT are abbreviations for four literal expressions. It should be noted that in the four words, only the first four letters are actually used by OSCAR. It is therefore possible to refer to RKPROGRAM as RKPR.

9b The instructions are written to evaluate the function F(X,Y); for the equation Y'=(X+Y)(X-Y) we use F(X,Y)=(X+Y)*(X-Y).

9c The starting point is chosen as X=0, Y=1 (precision 10); the last value of X as XLAST=1. With M=10, the solution will be printed for the interval 0.1 in X. The value of H is chosen smaller than this to ensure the accuracy of the calculation. With M=10 and H=0.01 the function is evaluated 10 times for each value printed.

9d The calculation may be extended by redefining XLAST and leaving X and Y unchanged. M is changed to 2, which changes the interval of the printout to 0.5. With H left unchanged, the function is evaluated 50 times for each value printed.

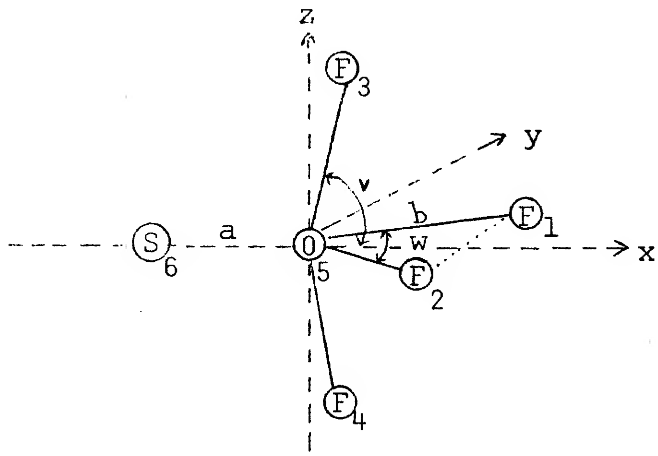
9e Under OS-3, each user is given 60 seconds of computer time. TIME CUT indicates that 60 seconds have been used. This is actual computer time, not time at the console. The statement TIME=120 extends the time to 120 seconds. The statement GO causes the computer to continue the calculation which was interrupted.

9f The same function as above is evaluated in the interval X=0 to X=-1, starting again at Y=1 and using H=-0.01.

9g To evaluate the set of two equations: $Y_1' = Y_2, Y_2' = -Y_1$ we define Y' as an array A with two components A(1) and A(2). The two equations are written as A(1)=Y(2) and A(2)=-Y(1) and F(X,Y) is defined as three statements enclosed in square brackets.

9h The starting point is chosen as X=0 with two values for Y: $Y_1 = \sin X = 0$, $Y_2 = \cos X = 1$ (precision 10). XLAST is $\pi/2$. H and M have the same values as in 9c.

EXAMPLE 10. Calculation of interatomic distances for a molecular model.



In this example, the calculation of interatomic distances is carried out for a model of the molecule SOF_4.

The geometry of SOF_4 is indicated in the diagram above. The cartesian coordinates of the atoms may be calculated from the three bond distances a, b, c, and the angles u and v using the following expressions:

$x_1 = b \cos u/2$	$y_1 = b \sin u/2$	$z_1 = 0$
$x_2 = x_1$	$y_2 = -y_1$	$z_2 = 0$
$x_3 = c \cos v$	$y_3 = 0$	$z_3 = c \sin v$
$x_4 = x_3$	$y_4 = 0$	$z_4 = -z_3$
$x_5 = 0$	$y_5 = 0$	$z_5 = 0$
$x_6 = -a$	$y_6 = 0$	$z_6 = 0$

The distances between two atoms i and j is calculated as:

$$DIS_{i,j} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2}$$

On the following pages, this calculation is carried out using OSCAR.

Example 10. Calculation of interatomic distances for a molecular model.

```
#job number,user identification  (CR)

#OSCAR  (CR)

OSCAR AT YOUR SERVICE V18 03/13/68  1633

*TO CALCULATE INTERATOMIC DISTANCES FOR SOF4  (CR)

DIS::='SQRT((X(I)-X(J))↑2+(Y(I)-Y(J))↑2+(Z(I)-Z(J))↑2)'  (CR)    }10a

FOR I=1 TO 6 DO[X(I)=0;Y(I)=0;Z(I)=0]  (CR)                      }10b

X(1)::='B*COS(U/2)'; Y(1)::='B*SIN(U/2)'  (CR)
X(2)::='X(1)';Y(2)::='-Y(1)'  (CR)
X(3)::='C*COS(V)';Z(3)::='C*SIN(V)'  (CR)                        }10c
X(4)::='X(3)';Z(4)::='-Z(3)'  (CR)
X(6)::='-A'  (CR)

A=1.405; B=1.535; C=1.595  (CR)                                  }10d
CONST=PI/180; U=120*CONST; V=81.8*CONST  (CR)

FOR I=1 TO 6 PRINT X(I),Y(I),Z(I),CR  (CR)
       0.7675         1.32935          0
       0.7675        -1.32935          0
       0.22749P5      0        1.57869                            }10e
       0.22749P5      0       -1.57869
       0       0      0
      -1.405          0        0

LET P(I,J) BE PRINT I,J,DIS,CR  (CR)                             }10f

I=1; FOR J=2 TO 6 DO P(I,J)  (CR)
       1       2      2.6587
       1       3      2.133315
       1       4      2.13332
       1       5      1.535
       1       6      2.546945
I=2; FOR J=3 TO 6 DO P(I,J)  (CR)
       2       3      2.133315
       2       4      2.13332
       2       5      1.535                                       }10g
       2       6      2.546945
I=3; FOR J=4 TO 6 DO P(I,J)  (CR)
       3       4      3.15738
       3       5      1.595
       3       6      2.27097
```

```
I=4; FOR J=5 TO 6 DO P(I,J)  CR                          10g
       4       5        1.595                            continued
       4       6        2.27097

DO P(5,6)   CR
       5       6        1.4051P5

U=125*CONST; DO P(1,6)   ESC   1   6   2.51435           10h
U=115*CONST; DO P(1,6)   ESC   1   6   2.57834
```

$\boxed{\text{Control Shift A}}$

```
#LOGOFF   CR
TIME   7.110 SECONDS.   MFBLKS 0000
#
```

Comments to Example 10

10a DIS is defined as the literal expression given on page 2.42.

10b Since many of the coordinates have the value zero, it is expedient to start by setting all coordinates equal to zero.

10c The coordinates with values different from zero are defined as the expressions given on page 2.42.

10d The bond distances A, B, and C are given numerical values. The angles U and V are given values in radians corresponding to 120° and 81.8°.

10e The cartesian coordinates of the atoms are calculated and printed.

10f Use of the function P simplifies the calculation and printing of the interatomic distances.

10g To calculate all interatomic distances it is necessary to let I assume the values 1 through 5, and for each value of I, give J the values I+1 through 6.

10h Here is shown the effect on a specific distance of change in one parameter. This, of course, can easily be expanded to include more distances and more parameters.

Example 11. Data file created in OSCAR then transferred to a Fortran program.

```
#EQUIP,10=FILE   CR
#OSCAR  CR

OSCAR AT YOUR SERVICE V36 08/07/68  1033

&OUTPUT,10   CR

&   ESC
FOR I=1 TO 100 PRINT (PI↑I)%11,(E↑I)%11,(2PI4↑I)%11,CR   CR      11a

&OUTPUT,10  CR

&  Control Shift A
#REWIND,10   CR                                                   11b
#SAVE,10=DAT  CR
#EDIT   CR                                                        11c

]TAPE  CR

      PROGRAM DEMO
      DIMENSION B(100,3)
1     FORMAT (I4,3E18.10)
      IC=0
      DO 2 I=1,100
      B(I,1)=READIN(10,IC)
      IF (IC.EQ.-1) GO TO 5
      B(I,2)=READIN(10,IC)
2     B(I,3)=READIN(10,IC)
      I=101
5     I=I-1
      DO 6 J=1,I
6     PRINT 1,J,B(J,1),B(J,2),B(J,3)                              11d
      END

      FUNCTION READIN(L,I)
      DIMENSION C(17)
1     IF (I.LT.0) I=0
      IF (I.EQ.0) READ(L,2)C
2     FORMAT (17A8)
      GO TO(4,3) EOFCKF(L)
3     READIN=SCANIN(C,I,136)
      IF (I.LT.0) GO TO 1
      RETURN
4     I=-1
      END
         FINIS
 ESC
```

```
  ESC
]OUT,DEMO    CR                                                   } 11e

]FORTRAN,I=DEMO,R    CR

 NO ERRORS FOR DEMO
                                                                  } 11f
 NO ERRORS FOR READIN
RUN



  1   3.1415926536E 00   2.7182818285E 00   2.0000000000E 00
  2   9.8696044011E 00   7.3890560988E 00   4.0000000000E 00
  3   3.1006276680E 01   2.0085536923E 01   8.0000000000E 00
  4   9.7409091035E 01   5.4598150032E 01   1.6000000000E 01
  5   3.0601968479E 02   1.4841315910E 02   3.2000000000E 01
  6   9.6138919357E 02   4.0342879349E 02   6.4000000000E 01
  7   3.0202932278E 03   1.0966331584E 03   1.2800000000E 02
  8   9.4885310158E 03   2.9809579870E 03   2.5600000000E 02
  9   2.9809099333E 04   8.1030839277E 03   5.1200000000E 02
 10   9.3648047475E 04   2.2026465795E 04   1.0240000000E 03
 11   2.9420401797E 05   5.9874141715E 04   2.0480000000E 03
 12   9.2426918151E 05   1.6275479142E 05   4.0960000000E 03  } 11g
 13   2.9036772706E 06   4.4241339201E 05   8.1920000000E 03
 14   9.1221711817E 06   1.2026042842E 06   1.6384000000E 04
 15   2.8658145969E 07   3.2690173725E 06   3.2768000000E 04
 16   9.0032220839E 07   8.8861105205E 06   6.5536000000E 04
 17   2.8284456359E 08   2.4154952754E 07   1.3107200000E 05
 18   8.8858240305E 08   6.5659969136E 07   2.6214400000E 05
 19   2.7915639496E 09   1.7848230096E 08   5.2428800000E 05
 20   8.7699567960E 09   4.8516519540E 08   1.0485760000E 06
 21   2.7551631843E 10   1.3188157345E 09   2.0971520000E 06
 22   8.65560041   BREAK    BREAK RELEASE
#
```

Comments to Example 11

11a The command &OUTPUT,10 will cause any following outputs to be written on logical unit ten. See &OUTPUT, page 3B.7; see also, Part One: Usage of Special Keys and Symbols (%).

11b The OS-3 control mode commands REWIND,LUN and SAVE, LUN=NAME are used to save the text under the name (in this case) DAT. (See Reference 5, pages 31.01 and 33.01.)

11c The OS-3 command mode, EDIT, was entered in order to input the Fortran program. The EDIT command TAPE allows the user to enter the program from a paper tape. (See Reference 4, page 41.01.)

11d The main program DEMO, enters data through subprogram READIN which employs a free format method. This allows the transfer of data from OSCAR, where data is printed in free form, to a Fortran program where data is confined to a specific format.

The function subprogram, READIN, calls SCANIN which is a free format BCD to floating point converter written by Walt Pawley. (See Reference 12.)

11e The EDIT command, OUT,DEMO, places the program in the file directory under the name DEMO. (See Reference 4, pages 28.01 through 29.01.)

The function subprogram, SCANIN, is available to all Fortran programs.

11f The OS-3 control mode command, FORTRAN,I=DEMO,R causes the program in file DEMO to be compiled and run. (See Reference 5, page 17.01.)

11g The program results were written in the specified format (I4,3E18.10). One hundred rows of data could have been listed (100 values of (PI↑I)%11, etc., were computed); however, execution was terminated by depressing the BREAK KEY.

[NOTE: OSCAR printout can occur in a format which indicates the precision. When this occurs, the procedure in this example cannot be used. For example, a Fortran program using SCANIN cannot use the following outputs from OSCAR:

```
I=1,
(PI↑I)%  3 is  3.14P3
(PI↑I)%  4 is  3.142P4
(PI↑I)%  5 is  3.1416P5

(PI↑I)% 15 is  3.1415926535898P15
(PI↑I)% 20 is  3.1415926535898P20
```

The usable range is % 6 to % 14:

```
(PI↑I)%  6 is  3.14159
(PI↑I)% 14 is  3.1415926535898.
```

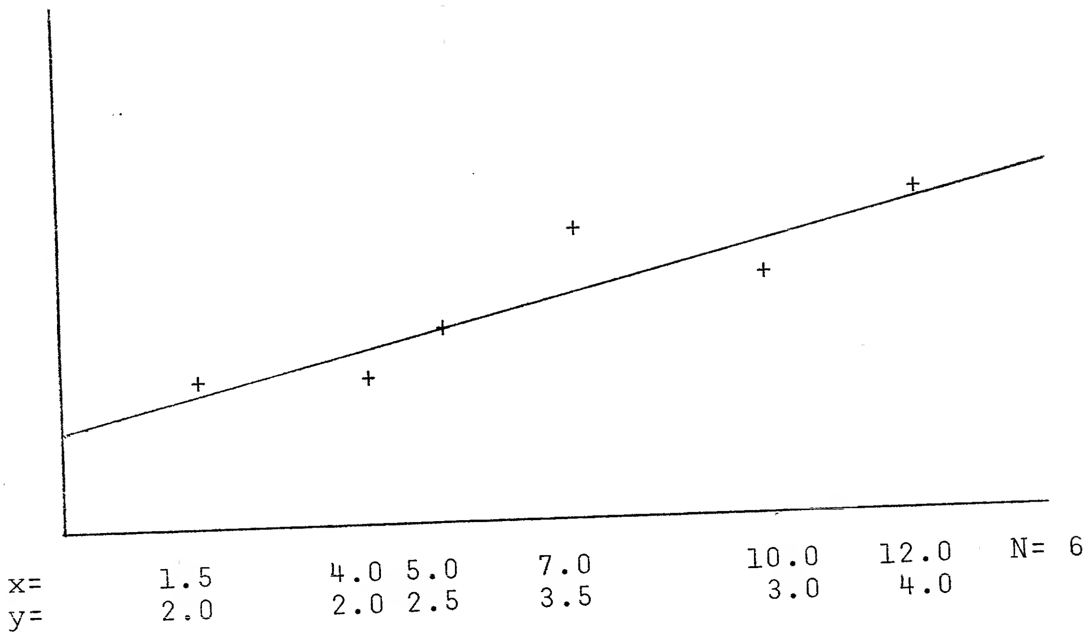
The FFIN function in the OS-3 system ignores the letter P without disrupting the input scan. (See Reference 9.)

6p The EQUIP statement assigns logical unit number 10 to the file named TEST. The routine COPY is used to get a printout of all the information on the file.

6q Under OSCAR, the information on file 10 may be read following the command &INPUT 10. The read statement includes eleven variable names. Only 10 suitable pieces of information are written on the file. M cannot be assigned any value.

6r This printout shows how the different variables have been assigned values from the file. A, B, C and F, G, H are single numbers. D and L are arrays. The first two texts written on the file as comments have been ignored. The last two lines of text have been read as alphanumeric strings to E and K. The date may serve as a useful label on the file. It is written in a form which cannot be read; it is neither a legal number nor a string in quotation marks. An attempt to read it as M caused an error statement and M is left undefined.

EXAMPLE 7. Least square test.



We have a set of N measurements which should fit the expression: $y = ax + b$

We measure x_i and y_i and want to determine a and b by least squares. The least squares criterion:

$$\sum_N (y_{meas} - y_{calc})^2 = \min$$

leads to the following expressions for a and b:

$$a = \frac{\Sigma x \Sigma y - N \Sigma xy}{(\Sigma x)^2 - N \Sigma x^2} \qquad b = \frac{\Sigma x \Sigma xy - \Sigma y \Sigma x^2}{(\Sigma x)^2 - N \Sigma x^2}$$

How this calculation may be carried out using OSCAR is shown on the following pages.

Example 7. Least squares test.

```
#job number,user identification   CR

#OSCAR   CR

OSCAR AT YOUR SERVICE V18 03/13/68  1605

* LEAST SQUARES TEST   CR

X=ARRAY(1.5 4. 5.7.10. 12.)   CR

ILLEGAL ARRAY CONSTANT                                                          7a

X=ARRAY(1.5 4. 5. 7. 10. 12.)   CR

Y=ARRAY(2. 2. 2.5 3.5 3. 4.)   CR

T=ARRAY(1 1 1 1 1 1)   CR

X*T IS  ESC    39.5                                                             7b
X*X IS  ESC   336.25
X*Y IS  ESC   126.0000

DEN::='(X*T)↑2-6*(X*X)';PRINT DEN   ESC   -457.25P5

A::='((X*T)*(Y*T)-6*(X*Y))/DEN';PRINT A   ESC          0.1848P5                 7c

B::='((X*T)*(X*Y)-(Y*T)*(X*X))/DEN';PRINT B   ESC      1.6167P5

YCALC::='A*X+B'; DIFF::='Y-YCALC'   CR                                          7d

FOR I=1 TO 6 PRINT X(I),Y(I),YCALC(I),DIFF(I),CR   CR
      1.5          2.          1.8939P5        0.1061P4
      4.           2.          2.3559P5       -0.3559P4
      5.           2.5         2.5407P5       -0.0407P3                         7e
      7.           3.5         2.9103P5        0.5897P4
     10.           3.          3.4647P5       -0.4647P4
     12.           4.          3.8343P5        0.1657P4

X=X%10;Y=Y%10   CR
                                                                                7f
PRINT A,B   ESC     0.184800437          1.61673045

FOR I=1 TO 6 PRINT X(I),YCALC(I),DIFF(I),CR   CR
      2P10         1.89393111       0.10606880
      2P10         2.35593220      -0.35593220                                  7g
      2.5P10       2.54073264      -0.04073264
      3.5P10       2.91033352       0.58966648
      3P10         3.46473483      -0.46473483
      4P10         3.83433570       0.16566430
```

$\boxed{\text{Control Shift A}}$

```
#LOGOFF   CR
TIME  5.796 SECONDS.  MFBLKS  0000
#
```

Comments to Example 7

7a The easiest way to carry out the calculation is to define X and Y as two vector arrays. Each element in the array must be separated by a blank space or a comma. The error message occurred because some spaces were omitted in the type in of X.

7b The array T is included to simplify the calculation ΣX. The two arrays T and X may be multiplied using the statement X*T. The result of this multiplication is the same as ΣX. According to the rules of matrix multiplication $X * X = \sum_i X_i * X_i$ and $X * Y = \sum_i X_i * Y_i$. These calculations are included only as a check.

7c A and B are defined as two functions according to the expressions given on page 2.31, replacing each summation by a vector multiplication. As the two functions have the same denominator, this is defined first and called DEN. The value of each function is calculated and printed. The elements of X and Y are decimal numbers assumed to have precision 6. A and B calculated from these numbers have precision lower than 6, indicated by P5. When a number with precision 5 is given with four figures only, as 0.1848P5, the next figure is a zero, as may be verified below.

7d YCALC is defined as the function A X + B and DIFF as the difference between Y and YCALC.

7e YCALC and DIFF are calculated and printed for each value of X.

7f The same calculation may be carried out using higher precision. The sign % is used to change the precision indicated for X and Y. As a result of changing the precision on X and Y to 10, A and B are calculated accurately to 9 figures.

7g YCALC and DIFF may be calculated with higher accuracy, also.

EXAMPLE 8. Newton's method for solving f(x)=0.

A solution to the equation f(x)=0 may be calculated from a trial value x_o to successively better approximation using the iteration formula

$$x_{n+1}=x_n-f(x_n)/f'(x_n) \quad n=0,1,2....$$

How OSCAR may be used to carry out this iteration process for a function of the form $f(x)=x^2-N$ is shown below.

```
#job number,user identification  CR
#OSCAR  CR
OSCAR AT YOUR SERVICE V18 03/13/68  1628                      } 8a
*TO SOLVE F(X)=0 BY NEWTON'S METHOD  CR
NEWT::='Y=X; X=X-F(X)/DF(X);PRINT Y,X,CR'  CR
LET F(X)=X↑2-N  CR                                            } 8b
LET DF(X)=2*X  CR
N=5; X=2.; Y=0  CR
WHILE X NEQ Y DO NEWT  CR
       2.               2.25
       2.25             2.23611
       2.23611          2.23607
       2.23607          2.23607
N=5; X=2P14; Y=0  CR                                          } 8c
WHILE X NEQ Y DO NEWT  CR
       2P14             2.2500000000000
       2.2500000000000                 2.2361111111111
       2.2361111111111                 2.2360679779158
       2.2360679779158                 2.2360679774998
       2.2360679774998                 2.2360679774998
```

Control Shift A

```
#LOGOFF  CR
TIME   3.724 SECONDS.  MFBLKS 0000
#
```

Comments to Example 8

8a NEWT is an abbreviation for a literal expression consisting of three statements separated by semicolons. The first gives Y the value of X_n, the second calculates X_{n+1} according to the iteration formula on the previous page, and the third prints the result.

8b The function $F(X) = X^2 - N$ is defined as the function F(X), and its derivative $F'(X) = 2X$ as DF(X). N is given the value 5, X_o is chosen as 2. Y may be given any value different from X_o.

8c The command: WHILE X NEQ Y DO NEWT (NEQ stands for "not equal to") causes the computer to compare X and Y; and to carry out the three statements of NEWT as long as the values for X and Y differ. X_{n+1} is calculated to a precision determined by the precision of X_o.
See NEQ, page 3A.17.

EXAMPLE 9: The Runge-Kutta Method for solving differential equations

Let the equation be $Y'=F(X,Y)$ with starting point (X_o,Y_o) and interval length H. We want to determine a constant K so that Y_o+K becomes as good an approximation of $Y(X_o+H)$ as possible. We use the following formula system:

$$\left.\begin{aligned} K_1 &= H\ F(X_o,Y_o) \\ K_2 &= H\ F(X_o+ 1/2H, Y_o+ 1/2K_1) \\ K_3 &= H\ F(X_o+ 1/2H, Y_o+ 1/2K_2) \\ K_4 &= H\ F(X_o+H, Y_o+K_3) \end{aligned}\right\} K = 1/6\ (K_1+2K_2+2K_3+K_4)$$

In this example, the instructions necessary to carry out the calculations have already been filed under the name RK. It remains for the user to define the equation he wants to evaluate, the starting point, the interval H, the last value of X, and a constant M, used to select the values of X for which the result will be printed (i.e. the result will be printed when $M \cdot X$ is a whole number.).

Two examples are used to illustrate these calculations. The first is an evaluation of the equation $Y'=(X+Y)(X-Y)$ over three different intervals. The second is an evaluation of a set of two equations: $Y_1'=Y_2, Y_2'=-Y_1$. A solution to these equations is: $Y_1=\sin X$, $Y_2=\cos X$; the calculation gives a table of sine and cosine values.

Example 9. The Runge-Kutta method for solving differential equations.

```
#job number,user identification    CR
#EQUIP,1=RK    CR
#OSCAR   CR

OSCAR AT YOUR SERVICE V18 04/12/68  1627

&CONTROL,1,TTY   CR
RKSTEP::='K1=H*F(X,Y);K2=H*F(X+H/2,Y+K1/2);K3=H*F(X+H/2,Y+K2/2);RK2'   9a
   RK2::='X=X+H;K4=H*F(X,Y+K3);Y=Y+SUM(K1,2*K2,2*K3,K4)/6'
   RKPROGRAM::='N=M  !(XLAST-X)/H!; FOR I=1 TO N DO STEPANDPRINT'
   STEPANDPRINT::='RKSTEP;IF FP(M*X)=0 OR I=N PRINT X%6,Y,CR'

LET F(X,Y)=(X+Y)*(X-Y)   CR
                                                                        9b
X=0; Y=1P10; XLAST=1; H=.01P10; M=10; PRINT X,Y,CR; DO RKPR   CR
       0       1P10
       0.1              0.909409365
       0.2              0.835785018
       0.3              0.777237583
       0.4              0.732726990
       0.5              0.701768385                                     9c
       0.6              0.684229362
       0.7              0.680176422
       0.8              0.689747815
       0.9              0.713041584
       1.               0.750015703

XLAST=4; M=2; PRINT X,Y,CR; DO RKPR   CR
       1.00000000000            0.750015703
       1.5              1.120270939                                     9d
       2.               1.679458983
       2.5              2.261564983
       3.               2.814215143
       3.5              3.346373000

TIME CUT
                                                                        9e
#TIME=120   CR
#GO   CR
       4.               3.868209405

X=0; Y=1P10; XLAST=-1; H=-.01P10; M=10;PRINT X,Y,CR; DO RKPR   CR
       0       1P10
      -0.1              1.110758849
      -0.2              1.246983600
      -0.3              1.417483014                                     9f
      -0.4              1.637325505
      -0.5              1.933662509
      -0.6              2.359549099
      -0.7              3.032347608
      -0.8              4.269162989
      -0.9              7.313295747
      -1.              26.685387916
```

Example 9. Continued

```
LET F(X,Y)=[A(1)=Y(2);A(2)=-Y(1);A]   (CR)                          } 9g

X=0; Y=ARRAY(0 1P10); H=.01P10; XLAST=PI/2; M=10; DO RKPR   (CR)
        0.1
 (0.09983341664   0.995004165    )
        0.2
 (0.19866933078   0.980066578    )
        0.3
 (0.29552020664   0.955336489    )
        0.4
 (0.38941834228   0.921060994    )
        0.5
 (0.47942553857   0.877582562    )
        0.6
 (0.56464247335   0.825335615    )
        0.7
 (0.64421768719   0.764842187    )                                  9h
        0.8
 (0.71735609085   0.696706709    )
        0.9
 (0.78332690958   0.621609968    )
        1.
 (0.84147098476   0.540302306    )
        1.1
 (0.89120736002   0.453596122    )
        1.2
 (0.93203908593   0.362357755    )
        1.3
 (0.96355818538   0.267498829    )
        1.4
 (0.98544972996   0.169967143    )

TIME CUT

#TIME=180   (CR)
#GO   (CR)
        1.5
 (0.99749498659   0.070737202    )                                  See 9e
        1.57
 (0.99999968293   0.000796327    )
```

Control Shift A

```
#LOGOFF   (CR)
TIME 122.167 SECONDS.   MFBLKS 0000
#
```

OSCAR: A User's Manual With Examples

PART THREE

SECTION A: FUNCTIONS AND CPERATORS

These words from the OSCAR language have been arranged in alphabetical order. Each word is given with its definition, examples of usage, brief explanatory notes regarding usage procedures and, in some cases, further references are included.

EXAMPLES OF FUNCTIONS AND OPERATORS

ABS The ABSOLUTE value function.

EXAMPLE 1 (arguments constant)

```
ABS 5 IS        ESC    5
ABS (-4) IS     ESC    4
ABS (+3. -4) IS   ESC    1.
ABS (-32.567) IS   ESC    32.567
```

EXAMPLE 2 (arguments variable)

```
X= -6; Y= +44.7378; ABS(X*Y) IS   CR
268.427
```

EXAMPLE 3 (arguments complex or array variables)

```
X=ARRAY(3. 4.)
!X! IS        5.
ABS(3. -4I) IS  ESC   5.000000
```

[NOTE: Complex numbers are treated by OSCAR as arrays.]

```
X=ARRAY(  (3.   3.)  ...
          (3.   3.) )
PRINT X,!X!  CR
```

[NOTE: ABS or the symbols !!]

```
(   (3.   3.   )
    (3.   3.   )   )       6.0000000
```

ACCEPT Same as READ. See page 3A.25 of this section.

AND A LOGICAL operator. See GTR, LSS.

EXAMPLE 1

```
A=4 GTR 3; B=3 LSS 5; A AND B IS   CR
            TRUE
```

EXAMPLE 2

```
X=4; Z=2.4; Y=3.1; W=1
IF X > Y AND Z < W THEN PRINT "YES" ELSE
PRINT Z
        2.4
```

EXAMPLE 3

```
A=TRUE; B=TRUE; A AND B IS   CR
TRUE

B=FALSE; A AND B IS  ESC    FALSE
A=-6.59 LSS -2.4    CR

A IS ESC    TRUE
A AND B IS   ESC    FALSE
B IS   ESC          FALSE
```

ARRAY This word defines an ARRAY constant. (See Reference 7, page 2, part II items 5 and 6.)

EXAMPLE 1

```
A=ARRAY(1 2 3 4 10)   CR
B=ARRAY(1 3 2 5 6)   CR
PRINT A+B   ESC   (2 5 5 9 16)
```

EXAMPLE 2

```
A=ARRAY((6.4 3.2 -7.59)  (0 1.4 ...
10)); Z=ARRAY ((6.4 3.2) (2.5 -2 ...
) (-4.5 11)); PRINT A*Z

( (83.115 -69.41 )
  (-41.5 107.20000 )  )
```

[NOTE: Three periods (...) will cause a carriage return and line feed without ending the line.]

ARCTAN This is the inverse tangent function.

BY

This identifies the increment value in a FOR statement.

(See Reference 7, page 4, D.)

EXAMPLE 1

```
LET A(J)=J*PI*2  (CR)
FOR J=10 BY 1 TO 15 PRINT J,A(J),J↑2,CR  (CR)
       10        62.831853071796       100
       11        69.115038378976       121
       12        75.398223686156       144
       13        81.681408993334       169
       14        87.964594300514       196
       15        94.247779607694       225
```

EXAMPLE 2

```
A=ARRAY((6.4 3.2 7)(0 1.4 10))  (CR)
FOR J=20 BY 4.5 UNTIL 35 PRINT J*A,J,CR  (CR)

((128.      64.       140    )
 (0     28.      200     )    )      20

((156.8     78.4      171.5000  )
 (0     34.3     245.0000   )      )       24.50000

((185.6     92.8      203.0000  )
 (0     40.6     290.0000   )      )       29.00000

((214.4    107.2      234.5000  )
 (0     46.9     335.0000   )      )       33.50000
```

CLEAR

This word frees previously defined variables. The emphatic assignment (::=) clears variables before assigning values to them. Clearing is not usually necessary; it need be used only when the previous content of a variable is a literal expression or a procedure. See pages 2.11, 2.12, item 4b.

(Also, See Reference 7, page 6, K.)

EXAMPLE 1

```
X=4; Y=5; CLEAR X  (CR)
PRINT X   ESC    []
```

[NOTE: Brackets mean undefined.]

EXAMPLE 2

```
A::=ARRAY(3I 3.2I 4.543)   CR
```

[NOTE: In this example, A is cleared before being set equal to ARRAY.]

```
B=A;CLAR\EA *
```

[NOTE: A line beginning with or ended by * is ignored. For this reason it can be used to erase a line with errors.]

```
B=A;CLA\EAR A; PRINT A,B   CR
```

[NOTE: Backslash, shift L, erases one character.]

```
[]
(0+3I 0+3.2I 4.543 )
```

EXAMPLE 3

```
J=4   CR
K=4; LET A(K)=K*PI*2   CR
PRINT A(K)   CR
        25.132741228718

CR=1.64   CR
A=CR*3   CR
PRINT A,CR   CR
PRINT ( A   )
     ARGUMENT IS PROCEDURE CALL
K    *PI    ARITHMETIC OPERAND (LEFT)
WHAT?

CLEAR A; A=CR*3   CR
PRINT A,CR   CR
        4.92       1.64
```

[NOTE: In this case, A must be cleared before it can be assigned another value.]

COS The cosine function. See pages 2.4-2.5, part 2c.

EXAMPLE 1

```
COS (1.) IS   ESC    0.54030P5
COS (-4.3) IS   ESC   -0.40080P5
```

```
COS (10*I) IS   ESC
COS ( TEMP )
     ARGUMENT IS WHAT?
```

[NOTE: Cosine of a complex number is presently undefined.]

CR This is a special variable. Initially, it has the meaning — carriage return and line feed. See pages 2.1 and 2.3, part 1g.

(See also, Reference 7, page 5 item g.)

EXAMPLE 1

```
A=3.14; B=2.6; PRINT "A=", A,CR,"B=",B,CR   CR
A=            3.14
B=            2.6
```

EXAMPLE 2

```
CR=1.64   CR
A=PI   CR
PRINT A,CR   CR
       3.1415926535898      1.64

PRINT A,CR, CR   CR
       3.1415926535898      1.64       1.64

CLEAR CR   CR
PRINT A,CR,A   CR
       3.1415926535898 []      3.1415926535898
```

DENOM The function for the denominator of a rational number.

EXAMPLE 1

```
DENOM(4/8) IS  ESC  8
```

DIV The integer division operator.

EXAMPLE 1

```
7 DIV 3    IS   ESC   2
```

DO This word causes the expression that follows it to be executed. See Example 5, page 2.15, items 5a and 5d. (See also, Reference 7, page 4, D.)

EXAMPLE 1

```
LET PR=PRINT A,B,C   CR
A=1; B=2; C=3.6   CR
DO PR   CR
   1    2    3.6
```

EXAMPLE 2 (See Reference 7, pages 7, 8, parts N and O.)

```
30.1: X=32; Y=48.999   CR
30.2: YX=Y*X; XX=X**2; PRINT YX, " YX"   CR
31.2: PRINT "XX= ",XX   CR

DO PART 30   CR
     1567.97    YX
DO STEP 31.2  CR
XX=           1024
```

E A special variable, E, the base of the natural logarithm system.

EXAMPLE 1

```
PRINT E    ESC    2.7182818284590
LN E IS    ESC    1.0000000000000
```

ELSE This word occurs in a conditional statement. (See Reference 7, page 4, C.)

EXAMPLE 1

```
Z=4.231; IF Z GTR PI THEN PRINT PI ELSE PRINT Z  (CR)
      3.1415926535898
Z=1.231; IF Z > PI THEN PRINT PI ELSE PRINT Z  (CR)
      1.231

Y=6I   (CR)
X='IF J < Y THEN PRINT I ELSE PRINT Y+Z'
0+1I
```

ENTIER   The function for the greatest integer less than or equal to the argument.

EXAMPLE 1

```
ENTIER (34.99999999) IS      34
ENTIER (44.000001) IS        44
ENTIER (-14.3) IS           -15
```

EP   The function for the exponent part of an expression.

EXAMPLE 1

```
X=360.   (CR)
EP X IS   (CR)
EP ( X  )
     OPERATION NOT IMPLEMENTED
```

[NOTE: NP X 10^{EP} = ARGUMENT
EP 360. = 2
NP 360. = 3.6
3.6 X 10^2 = 360]

EPRINT   The emphatic print. This command prints the actual form of the expression or quantity.

EXAMPLE 1

```
EPRINT 31435   (CR)
     31435
X::=PDL (2 3 18)   (CR)
EPRINT X   (CR)
     PDL ...
     (2 3 18)
```

[NOTE: For use of PDL, see page 3A.21.]

```
PRINT X   CR
    18
```

EQ A relational operator equal to. This command cannot be used to set a variable equal to a value. It is used only as a test for equality. See page 2.32, part 7c.

EXAMPLE 1

```
Y=TRUE; X=FALSE; Z=FALSE; Z EQ X IS   CR
TRUE
X EQ Y IS   ESC   FALSE
Y EQ Z IS   ESC   FALSE
```

EXP The exponential function E to the power. E is defined as 2.7182818284. (See Reference 7, page 8, item 2e.)

EXAMPLE 1

```
EXP(1) IS    ESC     2.7182818284604
EXP(34) IS   ESC     5.8346174252747E14
EXP(2) IS    ESC     7.3890560989341
E IS   ESC           2.7182818284604
```

EXPR If Y is a variable which represents a literal expression, then the statement Z=EXPR Y will transfer whatever Y contains to Z. The statement Z=Y would transfer the value of the expression to Z.

EXAMPLE 1

```
Y='PRINT A,B,C'   CR
X=EXPR(Y)   CR
A=1; B=2I; C=-17.3   CR
DO X   ESC
   1   0+2I    -17.3
DO Y   ESC
   1   0+2I    -17.3
```

```
Z=Y   CR
    1     0+2I    -17.3
Z←Y  ASSIGN WHAT?
```

EXAMPLE 2

```
X=79  CR
LET Y=X**2+34.45   CR
XX=EXPR(Y)  CR
PRINT XX  ESC     6275.4500
```

[NOTE: A print statement has no value.]

```
PRINT Y  ESC      6275.4500
```

FALSE The logical constant (Boolean). (See Reference 7, page 3., Expressions.)

EXAMPLE 1

```
X=3.1; Y=3.2; LET B1=X EQ Y   CR
B1 IS  CR
     FALSE
PRINT X,Y,B1   ESC   3.1   3.2   FALSE
Y=3.1  CR
PRINT X,Y,B1   CR
     3.1   3.1   TRUE
```

EXAMPLE 2

```
X=3.463; Y=3.6; LET Q=X NEQ Y   CR
LET R=X LSS Y  CR
Q AND R IS  ESC      TRUE
Q OR R IS  ESC       TRUE
Q XOR R IS  ESC      FALSE
Q IS  ESC       TRUE
R IS  ESC       TRUE
P = FALSE  CR
Q AND P IS  ESC      FALSE
Q OR P IS  ESC       TRUE
```

FOR The beginning of a FOR statement. See page 2.1, part 1g. (See also, Reference 7, page 4,D.)

EXAMPLE 1

```
FOR Z=5 BY 2 UNTIL 9 PRINT Z,Z↑2 CR   CR
      5        25
      7        49
      9        81
```

EXAMPLE 2

```
FOR Z=15.349 BY .2 UNTIL 15.9 PRINT Z,Z↑3,CR   CR
      15.349        3616.1
      15.549        3759.3
      15.749        3906.24
```

FP The function for the fraction part of a number. See IP.

EXAMPLE 1

```
FP (6.999) IS   CR
   .999P5

FP (-34.567) IS   ESC   -0.567P4
```

GCD The function for the greatest divisor of the arguments.

EXAMPLE 1

```
GCD (20,40,45) IS   ESC   5
```

GEQ A relational operator meaning greater than or equal to (>=). OSCAR will also accept the symbols >=.

EXAMPLE 1

```
X=10; Y=12; WHILE Y GEQ X SET X=X+1   CR

PRINT X   ESC   13
```

EXAMPLE 2

```
 9.0: Z=10.0   (CR)
 9.1: Y=E; X=0; PRINT Z   (CR)
10.2: Z=Z-1; X=2*Y*Z; PRINT "X",X,CR,Z,CR   (CR)
 DO PART 9   (CR)
      10.

 WHILE Z GEQ 7.5 DO STEP 10.2   (CR)
 X              48.929P5
        9P5
 X              43.493P5
        8P5
 X              38.056P5
        7P5
```

GO Same as GO TO and GOTO.

GOTO Used in stored programming only, to execute steps starting at a new step.

GTR A relational operator, greater than. OSCAR will accept the symbol > or the letters GTR.

EXAMPLE 1

```
X=4.1; Y=3.1; IF X GTR Y PRINT "X" ELSE ...
PRINT "Y"   (CR)
X
```

EXAMPLE 2

```
10.1: X=10; Y=102.39   (CR)
10.2: XX=X+Y; Y=2*XX   (CR)
11.6: PRINT "Y",Y   (CR)
11.7: SET Y=-1; DO PART 11.6   (CR)
DO PART 10   (CR)
IF Y GTR 219.19 DO STEP 11.6 ELSE DO STEP 11.7   (CR)
Y             224.78
```

I A special variable. I's initial value is 1I. 1I=SQRT(-1), and also 1I=1J.

EXAMPLE 1

```
SQRT(-1) IS  ESC  0+1I
```

EXAMPLE 2

```
X=32**2; Y=16.4+X*I; PRINT "Y=", Y  CR
Y=      16.4+1024I
PRINT Y*Y  ESC  -1.048307040E6 + 33587.2I
```

IF The beginning of a conditional statement or expression. (See Reference 7, page 4, C.)

EXAMPLE 1

```
Z=6; Q=5; IF Z GTR Q THEN Y=Z ELSE Y=Q; PRINT...
Y**2  CR
    43.56
```

[NOTE: GTR means greater than (>), See GTR.]

EXAMPLE 2

```
9.8: X=6*I; Y=6.00001*I; PRINT X,Y  CR
10.2: IF X EQ Y DO STEP 11.3 ELSE DO STEP 12.4  CR
```

[NOTE: EQ tests for equality. See EQ.]

```
11.3: X=X+12; PRINT X  CR
12.4: PRINT "NOT EQUAL  CR
DO PART 9  CR
  0+6I   0+6.00001I
DO PART 10  CR
NOT EQUAL
```

IM The function for the imaginary part of a complex number. See RE.

EXAMPLE 1

```
IM(12.3 + 2I) IS  ESC  2
```

IP — The function for the integer part of a number. See FP.

EXAMPLE 1

IP(6.9878) IS ESC 6

IS — A command which means print the value of the preceding expression.

EXAMPLE 1

3.45 + 24.9 + 23 IS ESC 51.35

J — A special variable initially equal to I. I is equal to SQRT(-1) and is used to represent complex numbers. See I.

KIND — The function indicating what kind of quantity the argument is. (See Reference 7, page A.8.)

EXAMPLE 1

KIND(SQRT(-1)) IS ESC COMPLEX

KIND(TRUE I\) IS ESC LOGICAL

A::=ARRAY(23 45 6.4)

KIND A IS ESC ARRAY

[<u>NOTE</u>: Logical constant TRUE; see TRUE, page 3A.29.]

[<u>NOTE</u>: Backslash, shift L, causes one character to be ignored.]

LEQ — A relational operator meaning — less than or equal to. OSCAR will also accept the symbols <=.

EXAMPLE 1

```
LET A=3 LEQ Y; Y=4; A IS   CR
TRUE
```

EXAMPLE 2

```
WL=3.495*63.4; ZL=32*PI*70.295  CR
IF ZL LEQ WL PRINT WL-ZL ELSE PRINT "NOT EQUAL  CR
NOT EQUAL
```

LET The beginning of a literal expression or procedure definition. See pages 2.1 and 2.3, part le.

(See also, Reference 7, page 5, H.)

EXAMPLE 1

```
Y=3; X=6I; LET F(X) = Y+X+2.3   CR
PRINT F(4)   ESC     9.3
PRINT F(X)   ESC    3+6I
PRINT X,Y,F(3.4)  ESC    0+6I    3    8.7
```

EXAMPLE 2

```
LET G(X,Y)=X**2+Y**2+329.14   CR
LET H(A,B)=A+B   CR
YYY='PRINT "G(X,Y)+H(A,B)",G(X,Y)+H(A,B),CR'   CR
```

[NOTE: YYY='PR ...'is a literal expression. The statement means make YYY an abbreviation for the expression 'PR ....']

```
CLEAR X,Y,A,B   CR
X=2.9; Y=3.4; A=1; B=3I   CR
DO YYY   CR
G(X,Y)+H(A,B)   350.11+3I
```

EXAMPLE 3

```
LET H BE PRINT X,Y,CR   CR
X=2.9; Y=3.4   CR
DO H   CR
     2.9      3.4
```

LN The function for the natural logarithm, i.e. base e.

EXAMPLE 1

```
LN(4) IS       ESC      1.3862943611240
LN 3.3 IS      ESC      1.193921

LN(-14.5) IS   ESC      2.676149 + 3.1415926535898I

LN E IS        ESC      1.0000000000000
PRINT E        ESC      2.7182818284590

LN(.0000000009) IS   CR
     -20.828626
```

LOG Same as LN.

EXAMPLE 1

```
LOG(4) IS    ESC      1.3862943611240
```

LOGT The function for base ten logarithm.

EXAMPLE 1

```
LOGT 4.2 IS    ESC        0.6232499
LOGT 4 IS      ESC        0.60205999132976

LOGT 10 IS   CR
       1.0000000000000
LOGT (-4.56) IS    ESC      .65896 + 1.3643763538419I

LOGT 100 IS    ESC        2.0000000000000
LOGT .0000000001 IS    ESC      -10.000001
```

LSS A relational operator meaning less than. OSCAR will also accept the symbol <.

EXAMPLE 1

```
X=3; Y=4; A='X LSS Y'; A IS  (CR)
 TRUE
```

EXAMPLE 2

```
9.0: Z::=10.0  (CR)
9.1: Y=E; *  (CR)
```

[NOTE: * causes the entire line to be ignored.]

```
9.1: Y=E; X=0; Z=Z-1  (CR)
10.2: Z=Z+1; X=2*Y*Z; PRINT "X",X,CR  (CR)

DO PART 9  (CR)
WHILE Z LSS 14 DO STEP 10.2  (CR)
X              54.3656
X              59.8022
X              65.2388
X              70.6753
X              76.1119
```

MAX The function for the maximum of arguments.

EXAMPLE 1

```
MAX(45,34,8) IS (ESC)  45
MAX(10I,I) IS (CR)
10I# MAX I
     OPERATION NOT IMPLEMENTED FOR ARRAYS
```

EXAMPLE 2

```
X=4.23; Y=15.04  (CR)
MAS *
```

[NOTE: * causes the entire line to be ignored.]

```
MAX(X,Y) IS  (CR)
     15.04

MAX(-89,  .98,23.34,567.45) IS  (ESC)  567.45
```

MOD The Modulus operator meaning — the remainder of the division.

[NOTE: Operation is not yet implemented.]

EXAMPLE 1

```
8 MOD 3 IS   ESC   2
```

[NOTE: 8 MOD 3 = 2
9 MOD 2 = 1]

NEQ A relational operator meaning — not equal to.

EXAMPLE 1

```
LET B=3 NEQ Y; Y=5  CR
    B IS  ESC      TRUE
```

EXAMPLE 2

```
Y=TRUE; X=FALSE; Z=FALSE; Z EQ X IS   CR
  TRUE
```

[NOTE: See EQ, page 3A.8 of this section.]

```
X NEQ Y IS   CR
  TRUE
X NEQ Z IS   ESC     FALSE
Y NEQ Z IS   ESC     FALSE
```

EXAMPLE 3

```
X=14.5; Y=15.04; PRINT Y    CR
    15.04
LET YYY=X*X + Y*X   CR
LET XXX=Y*Y + X*Y   CR
DO YYY; DO S\XXX   CR
```

[NOTE: The backslash, shift L, causes one character to be ignored.]

```
IF XXX NEQ YYY PRINT "NO SOLUTION",CR,"XXX",XXX,CR,...
"YYY",YYY   CR
NO SOLUTION
XXX            444.282
YYY            428.33
```

NOT A Unary logical operator, (the opposite).

EXAMPLE 1

```
LET A=4 GTR 3; A IS  CR
```

[NOTE: See GTR (>), page 3A.11 of this section.]

```
TRUE

NOT A IS  CR
FALSE
NOT FALSE IS  ESC      TRUE
```

EXAMPLE 2

```
X=14.5; Y=15.04  CR
LET B=X EQ Y  CR
```

[NOTE: See EQ, page 3A.8 of this section.]

```
PRINT "NOT B", NOT B,CR,"B",B  CR
NOT B      TRUE
B          FALSE
```

NP The function for the number part of a variable with an exponent (compliment of EP).

EXAMPLE 1

```
X=100  CR
NP (X) IS  ESC
```

[NOTE: NP(100) = 1
EP(100) = 2

$NP \times 10^{EP} = ARGUMENT$
$1 \times 10^{2} = 100$]

```
NP ( TEMP )
     OPERATION NOT IMPLEMENTED
```

NUM The function for the numerator of a rational number.

EXAMPLE 1

```
NUM(10/3) IS  ESC  10
```

O A special variable. The letter O has the initial value of zero.

EXAMPLE 1

```
PRINT O  ESC    0
```

OR A logical operator, the inclusive OR. (If one or the other is true, then the answer is TRUE.) (See Reference 7, page 3, Expressions.)

EXAMPLE 1

```
LET A=3 GTR 4; LET B=3 GTR 2   CR
A OR B IS    ESC    TRUE
A AND B IS   ESC    FALSE
```

[NOTE: See AND, page 3A.1 of this section, or see XOR, page 3A.31.]

```
A IS  ESC    TRUE
B IS  ESC    FALSE
```

EXAMPLE 2

```
Y=1.1; X=2.2; D=4.4; R=3.3   CR
LET Z = Y GTR X   CR
LET Q = DGTR R   CR
SYNTAX ERROR.  MISSING OPERATOR
```

[NOTE: Typing error, DGTR ... should be D GTR. The user must include space between variable and logical operator.]

```
LET Q = D GTR R   CR
LET P = R GEQ X   CR
```

[NOTE: See GTR (>), page 3A.11 and GEQ (>=), page 3A.10 of this section.]

EXAMPLE 2 (continued)

```
Z OR R \\P IS (ESC)     TRUE
Z OR Q IS (ESC)         TRUE
Q OR P IS (ESC)         TRUE
```

[NOTE: The backslash, shift L, causes one character to be ignored.]

```
Z IS (ESC)      FALSE
P IS (ESC)      TRUE
Q IS (ESC)      TRUE

Z AND P IS (ESC)      FALSE
```

PART This identifies the group to be executed.

EXAMPLE 1

```
9.1: X=1 (CR)
9.2: Y=2 (CR)
10.6: X=44.; PRINT "X",X,CR (CR)
10.7: Y=3.4 (CR)
10.9: X=6.9 (CR)

DO PART 9 (CR)
PRINT X,Y (CR)
       1      2
DO PART 10 (CR)
X          44.
PRINT X,Y (CR)
       6.9          3.4
```

EXAMPLE 2

```
30.1: X=32; Y=48.999 (CR)
30.2: YX=Y*X; XX=X**2; PRINT "YX",YX (CR)
31.2: PRINT "XX=",XX (CR)
31.3: PRINT "STEP 31.3 JUST EXECUTED (CR)

DO STEP 31.3 (CR)
STEP 31.3 JUST EXECUTED
DO PART 30 (CR)
YX              1567.97
DO PART 31 (CR)
XX=             1024 STEP 31.3 JUST EXECUTED
```

(See Reference 7, pages 7 and 8, parts N and O.)

PDL This defines push-down list constant. It can be used to create a storage Push List.

EXAMPLE 1

```
X=PDL(1 2 3 17.4); X IS  CR
17.4
POP X  CR
X IS  ESC      3
POP X  CR
X IS  ESC      2
POP X  CR
X IS  ESC      1
POP X  CR
X IS  ESC  []
```

[NOTE: Brackets mean undefined.]

EXAMPLE 2

```
X::=PDL(10I -16.54372 1932)  CR
X IS  ESC     1932
POP X ESC
X IS  ESC  -16.54372

PUSH X     CR
X IS  ESC     []
X=33  CR
```

[NOTE: See PUSH.]

```
11.3: Y=X*64.3; PRINT "X",X,CR,"Y",Y,CR  CR
DO STEP 11.3  CR
X          33
Y        2121.9
POP X   ESC
X IS   ESC  -16.54372
DO STEP 11.3  ESC
X           -16.54372
Y        -1063.76

POP X; DO STEP 11.3  CR
X        0+10I
Y        0+643.I
POP X; DO STEP 11.3  CR
```

EXAMPLE 2 (continued)

```
DOSTEP (   11.3   )
   X      *64.3      ARITHMETIC OPERAND (LEFT)
???
```

[NOTE: When X was popped the last time it left X undefined. This resulted in the question asked by the computer.]

```
X IS  ESC  []
```

PI   A special variable whose initial value is π.

EXAMPLE 1

```
PRINT PI   CR
       3.14159265358 98
```

EXAMPLE 2

```
LET CIRC=2*PI*R    CR
LET AREA=PI*R**2    CR
R=1; DO CIRC    CR
DO AREA   CR
PRINT AREA, CIRC, AREA*R*4/3    CR
       3.1415926535898        6.2831853071796 ...
       4.1887902047863
```

EXAMPLE 3

```
LET AREA=2*PI*R**2    CR
R=4; PRINT AREA
     100.53096491487
R=3.678645; PRINT AREA
      85.02676
R=1.992; PRINT AREA
      24.9321
```

POP   This word pops up each variable following. It is used to move backwards in a PDL List. See PDL, page 3A.21.

[NOTE: If X=PDL(3 4 17), then the value of X starts at the right most value and moves leftward one value each time it is popped.]

PREC The function which gives the precision of its argument. The symbol % indicates to the computer that the precision is going to be changed. See also &PRECISION, page 3B.8 of the next section.

EXAMPLE 1

```
X=PI      CR
PREC X IS      ESC       14
Y=X %10     CR
Y IS     ESC          3.141592654
PREC Y IS      ESC           10
```

EXAMPLE 2

```
Z=-1694.123456789    CR
PREC Z IS     ESC           13
Q=Z %8    CR

Q IS     ESC       -1694.1235
Q*Z IS     ESC                2.8700543E6
LET QQ=Q*Z     CR
QQ IS     ESC       2.8700543E6
PREC QQ IS     ESC          8
PREC (Q*Z) IS     ESC       8
PREC Q*Z IS     ESC     -13552.98765431
PREC Z IS     ESC           13
```

PRINT This word causes the values of the expressions that follow to be printed. The variable CR is often used in a print statement.

[NOTE: CR used in print statements causes line feed and carriage return. See CR, page 3A.5.]

EXAMPLE 1

```
A=6.1; B=4.3
PRINT A,"A",CR,"B",B
```

[NOTE: The print for A,"A", is hard to read. It would be better to ask for "A",A.]

```
          6.1A
B                     4.3
```

PROD The function for the product of the arguments.

EXAMPLE 1

```
PROD(1,2,3,4,5,6,7,8,9) IS  ESC   362880
```

EXAMPLE 2

```
PROD(7.5929, 63.192, -89I) IS  ESC   0-42703.2I
```

EXAMPLE 3

```
YY=PROD(10.29+8I, 16.13+2I, 9.7, -11.3)   CR
YY IS   ESC   -16439.1-16399.8I
```

PUSH This word means — push down each variable following. This allows the user to redefine a variable without losing the old value.

EXAMPLE 1

```
X=6.12; PUSH X   CR
X=34   CR
X IS   CR
         34
POP X; X IS   ESC          6.12
PUSH X; X=777   CR
X IS   CR
        777
POP X; X IS   ESC          6.12
POP X; X IS   ESC   []
```

(For example of POP, see page 3A.21.)

RE The function for the real part of a complex number.

EXAMPLE 1

```
RE(SQRT(-1)) IS   ESC   0
```

[NOTE: $\sqrt{-1}$ = 0+1I, RE($\sqrt{-1}$) = 0]

READ

This word means read in values for variables following. See page 2.28 of PART TWO, items 6h and 6i.

(See also, Reference 7, page 5, F.)

EXAMPLE 1

```
READ X,Y     CR
-3.5  4.7     CR
PRINT X,Y     CR
   -3.5       4.7
```

EXAMPLE 2

```
1.01:CLEAR Z; READ X      CR
1.02:Y=X     CR
1.03:Z='Y↑2+16*Y+4.5'      CR
1.04:PRINT "Z",Z,CR      CR
1.05:PRINT "SQRT(Z)",SQRT Z,CR      CR
1.06:PRINT "END OF PART ONE      CR

DO PART 1     CR
3   CR
Z              61.50000
SQRT(Z)         7.842194
END OF PART ONE

DO PART 1     CR
-89.76     CR
Z            6625.2
SQRT(Z)        81.39535
END OF PART ONE
```

SET

The beginning of an assignment statement (may be omitted in most situations).

EXAMPLE 1

```
FOR T=1 BY 2 UNTIL 5 SET A=T; PRINT A,CR     CR
       5
FOR T=1 BY 2 UNTIL 10 SET A=T*T; PRINT A,CR     CR
      81
FOR T=1 BY 2 UNTIL 10 SET A(T)=T*T;PRINT A,CR

(1  []  9  []  25  []  49  []  81)
```

[NOTE: Brackets mean undefined.]

EXAMPLE 2

```
20.0:Y=PI % 10  CR
20.1:WL=3.1415*14+Y  CR
20.2:ZL=4516.5  CR
21.3:IF ZL LEQ WL SET ZL=ZL+2*PI ELSE PRINT "NOT LEQ"
22.4:PRINT "ZL",ZL,CR,WL,"WL  CR
```

[NOTE: See LEQ, page 3A.13 of this section.]

```
DO PART 20
Y IS  ESC 3.141592654
ZL IS ESC 4516.5
DO PART 21,22   CR
SYNTAX ERROR.  ILLEGAL USE OF COMMA.

DO PART 21  CR
DO PART 22  CR
ZL          4516.5
      47.1226WL
```

[NOTE: For usage of PART, See Reference 7, page 7, N.]

SIGN   This is the function which gives the sign of its argument. The value is 1,0, or -1, depending on whether the argument is positive, zero, or negative.

EXAMPLE 1

```
SIGN(-67) IS  ESC   -1
SIGN(0) IS  ESC      0
SIGN(37.4) IS  ESC   1
```

SIN   The sin function in radians.

EXAMPLE 1

```
SIN 8.9 IS  ESC      0.50102P5
SIN I IS  ESC
SIN( I )
     ARGUMENT IS WHAT?
```

[NOTE: Sin of complex was not yet implemented at the time of this printing.]

EXAMPLE 1 (continued)

```
SIN(-14.2) IS (ESC) -0.99803P5
SIN(3) IS (ESC)     0.1411200080598672
```

SIZE A function; its value is an integer giving the size of the argument (the number of elements in a vector or the number of rows in an array).

EXAMPLE 1

```
X=PDL(16I 14.5 -32) (CR)
```

[NOTE: See PDL, page 3A.21.]

```
X IS (ESC) -32
SIZE X IS (ESC) 1
AA=ARRAY(1 2 3 4 5 8) (CR)
SIZE AA IS (ESC) 6

AB=ARRAY((32 42 -16.5)(172 2.3 4)) (CR)
SIZE AB IS (ESC) 2

BB=ARRAY(3 4 3.4 3.2 -17) (CR)
SIZE BB IS (ESC) 5
```

SQRT The square root function.

EXAMPLE 1

```
X=25; SQRT(X) IS (ESC) 5
SQRT(13.45+100I) IS (ESC) 7.561430+6.612505I
SQRT(-100) IS (ESC) 0+10I
SQRT(-14.2) IS (ESC) 0+3.76829I
```

STEP This identifies increment value in a FOR statement, or a step in a stored program.

EXAMPLE 1 (FOR statement)

```
FOR X=0 STEP 2 UNTIL 10 PRINT X*X,CR   CR
         0
         4
        16
        36
        64
       100
```

EXAMPLE 2 (STEP in a stored program)

```
40.1:X=32.8; Y=X**3; PRINT "40.1"   CR
40.2:PRINT X,Y   CR
40.3:LET U=X*Y,PRINT U   CR
40.4:LET ER=Y**3,PRINT "40.4",ER   CR
DO STEP 40.1   CR
        40.1
DO STEP 40.2   CR
        32.8      35287.6
DO STEP 40.3   CR
         1.15743E6
DO STEP 40.4   CR
        40.4      4.39405E13
```

STOP This word is reserved for future use.

SUM The function for the sum of arguments.

EXAMPLE 1

```
SUM(4, 5, 9, 34, 12) IS   ESC     64
SUM(2I, 34.5I, 56, 34) IS   ESC   90+36.5I
SUM(-19, 23.45, 67.9, -34) IS   ESC   38.35
```

TAN The tangent function.

EXAMPLE 1

```
TAN(4) IS   ESC   1.1578212823496
```

THEN — This word follows the IF clause and precedes the TRUE alternative in a conditional statement or expression.

EXAMPLE 1

```
Z=34; IF Z GTR 3.14 THEN SET X=3.14 ELSE ...
PRINT "Z",Z   CR

X IS   ESC     3.14
Z=2.1; IF Z > 3.14 THEN SET X=3.14 ELSE ...
PRINT "Z",Z   CR

Z       34
```

TO — This word identifies the final value in a FOR statement unless it follows the word GO. See FOR Statement, (See also, Reference 7, page 4, D.)

TRUE — A logical constant (Boolean Logic).

EXAMPLE 1

```
BA='3>4'  CR
BA IS  CR
  FALSE
CB=4>3  CR
CB IS  ESC  TRUE

DD=FALSE  CR
E=TRUE  CR

DD AND E IS  ESC    FALSE
DD OR E IS  ESC     TRUE
DD XOR E IS  ESC    TRUE
```

[NOTE: See XOR, OR, and AND.]

TTY The abbreviation for teletypewriter. This command information to be printed out from file onto teletypewriter unit (See &CONTROL, page 3B.3.)

TYPE Same as PRINT.

UNTIL This word identifies the final value in a FOR statement. See FOR, page 3A.9 and 3A.10.
(See also, Reference 7, page 4, D.)

EXAMPLE 1

```
FOR T=2 BY 2.5 UNTIL 5 PRINT T**2    CR
      4       20.25
```

VALUE The function for the value of an argument.

EXAMPLE 1

```
NEWT=6.59    CR
VALUE(NEWT ) IS    ESC    6.59
X='A+B*C'   CR
A=1.2; B=3.5; C=4.5   CR
X IS  CR
     16.95
VALUE *
VALUE X IS   CR
     16.95
PRINT VALUE ('A+B*C')   CR
     16.95
```

WHILE The beginning of a WHILE statement. (See Reference 7, page 4, E.)

EXAMPLE 1

```
100.1:X=2.3; Y=4.7; PRINT X,Y  (CR)

101.1:K=K+1; LET SS=X 2+Y; X=X+1; PRINT ...
X,Y,SS,CR  (CR)

102.1:WHILE X LSS Y DO STEP 101.1  (CR)

104.1:LET A(K)=X 2+5.3*Y+14*Y*X; PRINT K, ...
CR,A(K),X; X=X+1:K= ...
K+1  (CR)

104.2:WHILE X LEQ Y DO STEP 104.1  (CR)
```

[NOTE: See LSS, page 3A.15 of this section.]

```
DO STEP 100.1  (CR)
       2.3          4.7

K=0  CR

DO STEP 102.1  (CR)
       3.3          4.7          15.59
       4.3          4.7          23.19
       5.3          4.7          32.79

DO STEP 100.1  (CR)
       2.3          4.7
K=1  (CR)

DO STEP 104.2  (CR)
       1
     181.54         2.3          2
     252.94         3.3          3
     326.34         4.3
```

WRITE Same as PRINT.

XOR Logical operator; the exclusive or. (If only one is true then answer is TRUE. All other cases are FALSE.) See OR, page 3A.19.

EXAMPLE 1

[NOTE: A is a true statement. B is a false statement.]

```
A=3>2; B=5>6    CR
A IS    ESC      TRUE
B IS    ESC      FALSE
C= FALSE   CR

C AND B IS  ESC   FALSE
A OR B IS  ESC    TRUE
A XOR B IS  ESC   TRUE
A AND B IS  ESC   FALSE
C XOR B IS  ESC   FALSE
```

OSCAR: A User's Manual With Examples

PART THREE

SECTION B: COMMANDS

The commands used under OSCAR are identified by the ampersand (&) typed by the user, or by OSCAR. The commands have been arranged in alphabetical order with examples and explanations of their usage.

Depressing ESC will cause the & to be printed when the register is empty. Depressing ESC from the & mode will cause a return to the regular mode.

[NOTE: See Reference 7, Appendix F, for a complete listing of & commands.]

EXAMPLES OF COMMANDS

&BKSP,LUNLIST or
&BKSPACE,LUNLIST

This command causes each logical unit listed to be backspaced one record.

EXAMPLE:

```
#(account number),(user identification) CR
#EQUIP,3=DATA  CR
```

[NOTE: DATA is a saved file this user had entered previously. Each line corresponds to a record.]

```
#OSCAR  CR

OSCAR AT YOUR SERVICE V36 08/06/68  0840
ESC
&INPUT,3,TTY  CR

&  ESC
```

[NOTE: ESC Key in Oscar gives you & command mode. If you are in the & mode the ESC key returns you to the DIRECT mode. The user may type & directly as in &BKSP,3.]

```
READ A,B,C,D,E,F,G,H,I,J,K  CR
1234      6789      9876      34.678
6527      2497      2514      89.341
4634      7583      3580      23.567

&BKSP,3  CR

READ A  CR
4634      7583      3580      23.567

A IS  ESC   4634

&REWIND,3  CR

READ A  CR
1234      6789      9876      34.678

A IS  ESC   1234
&FWSP,3  CR

READ A  CR
4634      7583      3580      23.567
```

[NOTE: Each READ A redefines variable A. B,C,D will remain as originally defined until redefined.]

```
A IS  ESC  4634
B IS  ESC  6789
C IS  ESC  9876
D IS  ESC    34.678
```

[NOTE: For an example of READ, see page 3A.25 of the previous section.]

&CONTROL,LUN,TTY

This command causes statements or commands from (LUN) to be read in. TTY (optional) causes the text to be printed on the teletypewriter as it is read.

EXAMPLE: 1

```
#user number,identification
#EDIT
```

[NOTE: For EDIT commands, see Reference 4, Input,Out.]

```
]INPUT
00001:LET ZZZ=987654.23+X↑3
00002:X=2
00003:PRINT ZZZ,X,CR
00004:X=4
00005:PRINT ZZZ,X,CR
00006:
]OUT,ABC
```

[NOTE: The command OUT,NAME will save the information under the name you specify.]

```
#EQUIP,3=ABC   CR

]OSCAR

OSCAR AT YOUR SERVICE V36 08/05/68  1050
```

[NOTE: For OS-3 control commands, see Reference 5, EQUIP.]

```
&CONTROL,3,TTY   CR
```

&CONTROL (continued)

```
LET ZZZ=987654.23+X↑3
X=2
PRINT ZZZ,X,CR
   987662.23          2
X=4
PRINT ZZZ,X,CR
   987718.23          4
```

[NOTE: The above seven lines are all printed by the computer.]

```
X=12.5; PRINT ZZZ   CR
   989607.36
X=-67.89+10I   CR

PRINT ZZZ   CR
   695113.+137271.5I
```

[NOTE: User may type & sign or hit ESC to get the & sign.]

```
&REWIND,3   CR

&CONTROL,3   CR
   987662.23          2
   987718.23          4
```

EXAMPLE: 2

```
#EDIT

]INPUT
00001:LET F=SIN(X↑2)+COS(Y↑2)+SIN(X)
00002:LET G=34*X↑3+24*X 2+Y
00003:PRINT "F=",F,CR,"G=",G,CR
00004:IF F<G SET A=TRUE ELSE SET A=FALSE
00005:PRINT A
00006:

]OUT,EXAMPONE

] Control Shift A

#EQUIP,33=EXAMPONE
#OSCAR

OSCAR AT YOUR SERVICE V35 07/17/68  0939

X=4.5; Y=10
&CONTROL,33
F=        0.87031P5
G=     3594.25
   TRUE
```

&CONTROL (continued)

```
X=4.5; Y=10 (CR)
&REWIND,33 (CR)
&CONTROL,33,TTY (CR)
```

[NOTE: By typing TTY after &CONTROL,33, the user can have the file printed out as it is being used.]

```
LET F=SIN(X↑2)+COS(Y↑2)+SIN(X)
LET G=34*X↑3+24*X↑2+Y
PRINT "F=",F,CR,"G=",G,CR
F=        0.87031P5
G=     3594.25
IF F<G SET A=TRUE ELSE SET A=FALSE
EPRINT A
   TRUE
```

&DATE

This command causes the current date and time to be printed out.

EXAMPLE:

```
&DATE
07/30/68  1249

&OUTPUT,7
&DATE
```

[NOTE: This puts date on logical unit 7.]

&FWSP,LUNLIST or
&FWDSPACE,LUNLIST

This command causes each logical unit listed to be spaced forward one record.

FOR THIS EXAMPLE, PLEASE REFER TO PAGE 3B.1, &BKSP,LUNLIST or &BKSPACE,LUNLIST.

&INPUT,LUN,TTY or
&INPUT,LUN or
&INPUT,TTY

This command allows data constants to be read from a file by READ statements. If an

end-of-data is reached, data can be read from the teletypewriter (abbreviated TTY).

EXAMPLE:

```
#EDIT   CR

]INPUT   CR
```

[NOTE: This is the EDIT command INPUT. Do not confuse it with the OSCAR command &INPUT, below.]

```
00001:1234    6789    9876    34.678
00002:6527    2497    2514    89.341
00003:4634    7583     3580   23.567
00004:
]OUT,DATA   CR
#EQUIP,1=DATA    CR
#OSCAR

OSCAR AT YOUR SERVICE V36 08/06/68  0833

&INPUT,1,TTY    CR

&    ESC
READ A,B,C,D    CR
1234    6789    9876    34.678

READ E,F,G,H    CR
6527    2497    2514    89.341

READ I   CR
4634    7583    3580    23.567

A IS   CR
    1234
B IS   ESC    6789
C IS   ESC    9876
D IS   ESC      34.678
I IS   ESC    4634
I IS   ESC    4634
J IS   ESC    0+IJ

&REWIND,1    CR

&INPUT,1    CR

&    ESC
READ Z,X,C,V    CR
```

&INPUT,LUN,TTY (Continued)

```
READ B,N,M,A,S,D,F,G   (CR)

Z IS  (ESC)   1234
X IS  (ESC)   6789
C IS  (ESC)   9876
V IS  (ESC)     34.678
G IS  (ESC)     23.567
F IS  (ESC)   3580
```

&OCTOUT,VARIABLE NAME,
VARIABLE NAME,...   This command causes the values of the specified variables to be printed in octal form.

EXAMPLE 1

```
#OSCAR   (CR)
OSCAR AT YOUR SERVICE V36 08/06/68  0900
Q=1   (CR)
(ESC)
```

[NOTE: ESC will cause the & to be printed.]

```
&OCTOUT,Q   (CR)
Q
20000634 >
10002002 INT
 00000001
&   (ESC)
W=2   (CR)
(ESC)
&OCTOUT,W   (CR)
W
20000646 >
10002002 INT
 00000002
&   (ESC)
Y=8   (CR)
(ESC)
&OCTOUT,Y   (CR)
Y
20002314 >
10002002 INT
 00000010
```

&OCTOUT,... (continued) EXAMPLE: 2

```
B=ARRAY(1 6 7 8 9 )    CR
ESC
&OCTOUT,B   CR
B
20000660 >
74022001 ARAY
  ( 20000646 >
    10002002 INT
     00000001 ,
    20000655 >
    10002002 INT
     00000006 ,
    20002304 >
    10002002 INT
     00000007 ,
    20002306 >
    10002002 INT
     00000010 ,
    20002310 >
    10002002 INT
     00000011  )
```

&OUTPUT,LUN,TTY or
&OUTPUT,LUN or
&OUTPUT,TTY

This command causes any following outputs (from print statements) to be written on the logical unit specified or on the teletypewriter until another &OUTPUT command is given.

EXAMPLE:

```
#OSCAR   CR
OSCAR AT YOUR SERVICE V36 08/06/68  0848
| Control Shift A |
#EQUIP,4=FILE   CR
#MI   CR
INTERRUPTED
```

[NOTE: OSCAR is left temporarily to equip logical unit 4 to a file.]

&OUTPUT,... (continued)

```
&OUTPUT,4,TTY   CR
```

[NOTE: For OS-3 control mode commands, EQUIP,MI, see Reference 5.]

```
&   ESC
PRINT "READ X   CR
READ X

PRINT "Y=X↑2   CR
Y=X↑2

PRINT "PRINT X,CR   CR
PRINT X,CR

PRINT "PRINT Y,CR   CR
PRINT Y,CR

&OUTPUT,4,TTY   CR

& Control Shift A
#REWIND,4   CR
#SAVE,4=XY   CR
#MI   CR
INTERRUPTED
```

[NOTE: For OS-3 control mode commands, REWIND,SAVE, see Reference 5.]

```
&CONTROL,4,TTY   CR
READ X
-897.654+10I   CR
Y=X↑2
PRINT X,CR
-897.654+10I
PRINT Y,CR
805683.-17953.08I

&REWIND,4   CR

&CONTROL,4   CR
-12.4   CR
    -12.4
    153.76
```

&PRECISION,<INTEGER> This command causes the standard precision of six to be changed to the

integer specified. See PREC, page 3A.23 of the previous section.

EXAMPLE:

```
#OSCAR  CR

OSCAR AT YOUR SERVICE V36 08/06/68  0909

X=34.56; Y=78.90  CR

X*Y IS  ESC   2726.78
PREC (X*Y) IS  ESC   6
&PRECISION,20  CR

&  ESC
X=34.56; Y=78.90  CR

X*Y IS  ESC   2726.784P20
&PRECISION=30  CR

&  ESC
X=34.567; Y=78.909  CR

X*Y IS  ESC   2727.647403P30
```

&PROGRAM,N,.M

This command is used for creating stored programs. It causes the line number, starting with the one specified, to be printed for each line.

EXAMPLE:

```
&PROGRAM,6,.02  CR

6.02:LET Q=PI*R↑2  CR
6.04:LET R=.5*D  CR
6.06:READ D  CR
```

[NOTE: See READ, page 3A.25.]

```
6.08:C='PI*D'  CR
6.10:PRINT Q,CR,R,CR,D,CR,C,CR  CR
6.12:PRINT "END OF PART ONE  CR
```

&PROGRAM N,.M(continued)

```
6.14:  ESC
DO PART 6   CR

5  CR
    19.635
     2.5
     5
    15.707963267949
END OF PART ONE
CLEAR C
DO PART 6
-12.7
    126.677
     -6.35
    -12.7
    -39.8982
END OF PART ONE
```

[NOTE: See Example 3, Stored Program, page 2.7 of this manual.]

&RESTART

This command clears the user's storage area.

EXAMPLE:

```
&RESTART   CR

OSCAR AT YOUR SERVICE V36 08/01/68  1428
```

&REWIND,LUNLIST

This command causes the LUN specified (file number between 0-99) to be rewound. This command is used after the contents of a file have been read or written. See &CONTROL, page 3B.2.

&UDUMP,LUN

This command causes one block to be written containing the contents of the user's OSCAR

storage area. This is for saving the status of OSCAR for a short time only. See example for &ULOAD, below.

&ULOAD,LUN

This command causes one block to be read from the LUN specified, restoring OSCAR to what it was when dumped. Presently, UDUMP and ULOAD must be used with the same version of OSCAR, V36 in this example.

EXAMPLE:

```
#EQUIP,11=FILE
#OSCAR

OSCAR AT YOUR SERVICE V36 08/01/68  1414

Q=21; R=-9.876354+10.4I
W='Q↑2+R*7+SQRT R+SIN Q'

&UDUMP,11

&[Control Shift A]
#SAVE,11=UDAT
#LOGOFF
TIME 2.340 SECONDS  MFBLKS 41 COST $0.25
```

[NOTE: LOGON numbers are blocked out by computer.]

```
#job number,user identification (CR)
#EQUIP,88=UDAT
#OSCAR
```

&ULOAD,LUN (continued)

```
OSCAR AT YOUR SERVICE V36 08/01/68  1423

&ULOAD,88

&   ESC
Q IS   ESC    21

PRINT W
 374.19650+76.2799I
R↑2 IS    -10.618P5-205.428I
```